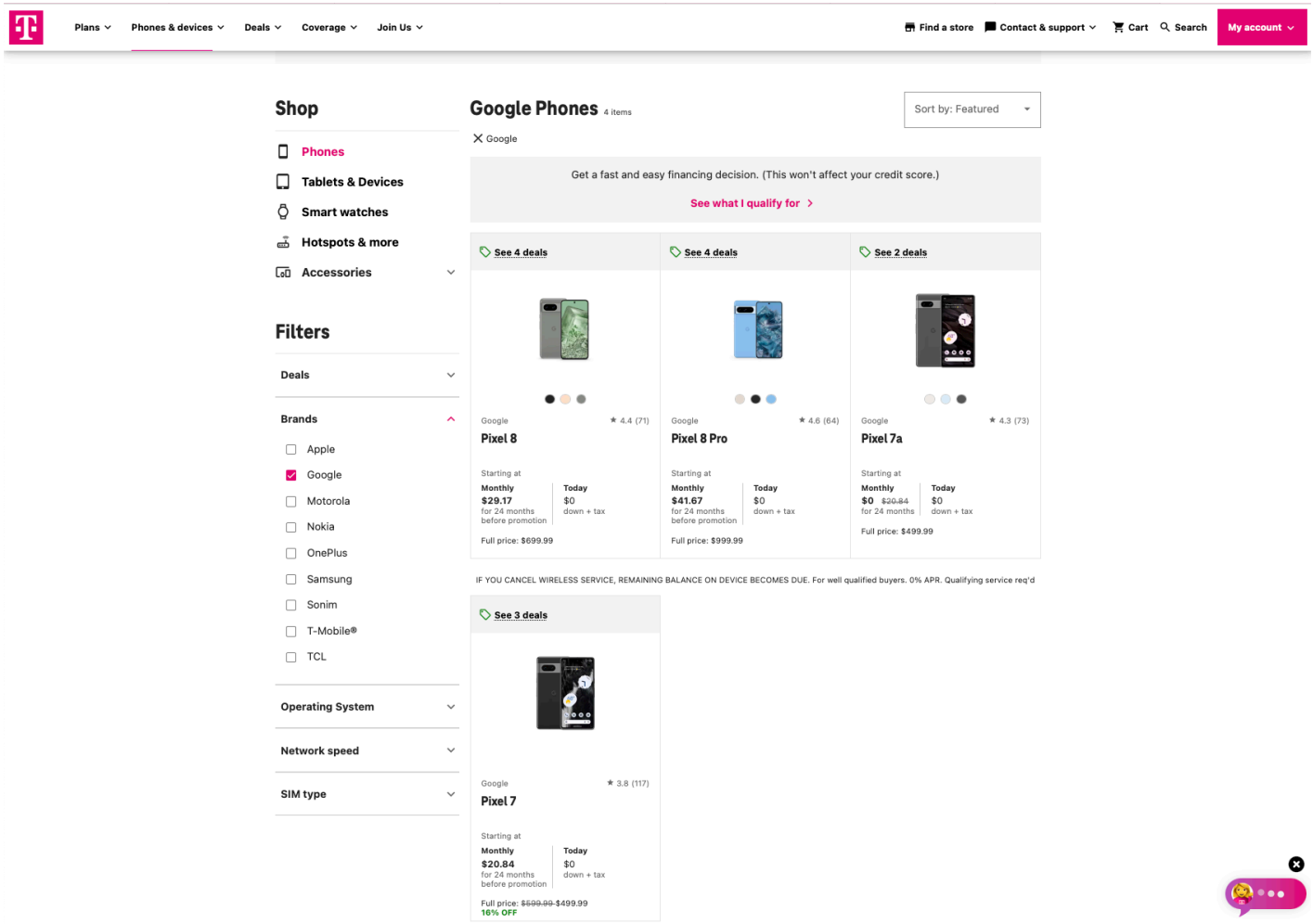
Exhibit L

Exhibit L - U.S. Patent No. 9,215,613 ("'613 Patent")

Accused Instrumentalities: smartphones, basic phones, tablets, laptops, and hotspot devices sold (including those sold in bundles with data plans) or used by T-Mobile and all versions and variations thereof ("Accused Instrumentalities") since the issuance of U.S. Pat. No. 9,215,613 (the "Asserted Patent").

Claim 1

Claim	Public Documentation
[1pre] A wireless end-user device, comprising:	The Accused Instrumentalities include "A wireless end-user device, comprising." For example, T-Mobile sells and uses devices described by T-Mobile's website below (e.g., devices made by Samsung, Apple, Motorola, Google, and Kyocera). These devices constitute a wireless end-user device as described in claim 1. *See, e.g.* https://www.t-mobile.com/cell-phones

Claim	Public Documentation
	 ; *see also* https://www.t-mobile.com/tablets; https://www.t-mobile.com/smart-watches; https://www.t-mobile.com/hotspots-iot-connected-devices.

Claim	Public Documentation
	As a specific example, Google's devices, including the Google Pixel 7, are wireless end-user devices which run the Android Operating System, and include a processor. *See, e.g.*, https://support.google.com/pixelphone/answer/7158570?visit_id=638458852688642944-1705659042&p=specs&rd=1 ; https://www.t-mobile.com/cell-phone/google-pixel-7:

Operating system	Android 13

Display	• Full-screen 6.3-inch (160.5 mm)[1] display
	• 20:9 aspect ratio
	• FHD+ (1080 x 2400) OLED at 416 PPI
	• Smooth Display (up to 90 HZ[2])
	• Always-on display
	  • At a Glance
	  • Now Playing

Dimensions and Weight[3]	**Dimensions** • 6.1 height x 2.9 width x 0.3 depth (inches) • 155.6 height x 73.2 width x 8.7 depth (mm) **Weight** • 197 g • 6.9 oz
Battery	• Beyond 24-hour battery life[4] • Up to 72-hour battery life with Extreme Battery Saver[4] • Minimum 4270 mAh • Typical 4355 mAh[5]
Memory and Storage	**Memory** • 8 GB LPDDR5 RAM **Storage** • 128 GB / 256 GB UFS 3.1 storage[9]
Processors	• Google Tensor G2 • Titan M2™ security coprocessor

Claim	Public Documentation

Additional spec details

Battery Description	4355 mAh
Ports	USB Type-C
Connectivity	Wi-Fi 6E (802.11ax) with 2.4GHz+5GHz+6GHz, HE160,MIMO, Bluetooth 5.2, NFC
Processor	Google Tensor G2
Operating System	Android
Ram	8 GB
Maximum Expandable Memory	0 GB
Wireless Network Technology Generations	4G LTE, 5G
Supported Email Platforms	GMail, Apple Mail, POP3, IMAP4, SMTP, Microsoft® Exchange, AOL, AIM, Yahoo!® Mail
Hearing Aid Compatibility	M3, T3
WEA Capable	true
Mobile Hotspot Capable	true
Frequency	GSM: 850 MHz, 900 MHz, 1800 MHz, 1900 MHz; CDMA: BC0, BC1, BC10; LTE: 1, 2, 3, 4, 5, 7, 8, 12, 13, 14, 17, 18, 19, 20, 25, 26, 28, 29, 30, 32, 38, 39, 40, 41, 42, 46, 48, 66, 71; UMTS: Band I (2100), Band II (1900), Band IV (1700/2100), Band V (850), Band VIII (900); 5G: n1, n2, n3, n5, n7, n8, n12, n14, n20, n25, n28, n30, n38, n40, n41, n48, n66, n71, n75, n76, n77, n78
Weight	6.9 Ounces
Length	0.3
Height	6.1
Width	2.9

What's in the box

- Google Pixel 7 Device
- 1m USB-C to USB-C cable (USB 2.0)
- Quick Switch Adapter
- SIM Tool

For WEA capability, see T-Mobile WEA

California residents: see the California Proposition 65 **WARNING**

Claim	Public Documentation
[1a] a wireless wide area network (WWAN) modem to communicate data for Internet service activities between the device and at least one WWAN, when configured for and connected to the WWAN;	The Accused Instrumentalities include "a wireless wide area network (WWAN) modem to communicate data for Internet service activities between the device and at least one WWAN, when configured for and connected to the WWAN." This WWAN modem in the Accused Instrumentalities provides a connection to a T-Mobile's wireless network. For example, Google's devices, including the Google Pixel 7 Pro, comprise a wireless modem for communicating with mobile service base stations such as a 4G LTE or 5G modem. *See, e.g.*, https://support.google.com/pixelphone/answer/7158570?visit_id=638458852688642944-1705659042&p=specs&rd=1 ; https://www.t-mobile.com/cell-phone/google-pixel-7:

Operating system	Android 13

Display	• Full-screen 6.3-inch (160.5 mm)[1] display
	• 20:9 aspect ratio
	• FHD+ (1080 x 2400) OLED at 416 PPI
	• Smooth Display (up to 90 HZ[2])
	• Always-on display
	  • At a Glance
	  • Now Playing

Dimensions and Weight[3]	**Dimensions** • 6.1 height x 2.9 width x 0.3 depth (inches) • 155.6 height x 73.2 width x 8.7 depth (mm) **Weight** • 197 g • 6.9 oz
Battery	• Beyond 24-hour battery life[4] • Up to 72-hour battery life with Extreme Battery Saver[4] • Minimum 4270 mAh • Typical 4355 mAh[5]
Memory and Storage	**Memory** • 8 GB LPDDR5 RAM **Storage** • 128 GB / 256 GB UFS 3.1 storage[9]
Processors	• Google Tensor G2 • Titan M2™ security coprocessor

Claim	Public Documentation

Additional spec details

Battery Description	4355 mAh
Ports	USB Type-C
Connectivity	Wi-Fi 6E (802.11ax) with 2.4GHz+5GHz+6GHz, HE160,MIMO, Bluetooth 5.2, NFC
Processor	Google Tensor G2
Operating System	Android
Ram	8 GB
Maximum Expandable Memory	0 GB
Wireless Network Technology Generations	4G LTE, 5G
Supported Email Platforms	GMail, Apple Mail, POP3, IMAP4, SMTP, Microsoft® Exchange, AOL, AIM, Yahoo!® Mail
Hearing Aid Compatibility	M3, T3
WEA Capable	true
Mobile Hotspot Capable	true
Frequency	GSM: 850 MHz, 900 MHz, 1800 MHz, 1900 MHz; CDMA: BC0, BC1, BC10; LTE: 1, 2, 3, 4, 5, 7, 8, 12, 13, 14, 17, 18, 19, 20, 25, 26, 28, 29, 30, 32, 38, 39, 40, 41, 42, 46, 48, 66, 71; UMTS: Band I (2100), Band II (1900), Band IV (1700/2100), Band V (850), Band VIII (900); 5G: n1, n2, n3, n5, n7, n8, n12, n14, n20, n25, n28, n30, n38, n40, n41, n48, n66, n71, n75, n76, n77, n78
Weight	6.9 Ounces
Length	0.3
Height	6.1
Width	2.9

What's in the box

- Google Pixel 7 Device
- 1m USB-C to USB-C cable (USB 2.0)
- Quick Switch Adapter
- SIM Tool

For WEA capability, see T-Mobile WEA

California residents: see the California Proposition 65 **WARNING**

Claim	Public Documentation
[1b] a wireless local area network (WLAN) modem to communicate data for Internet service activities between the device and at least one WLAN, when configured for and connected to the WLAN;	The Accused Instrumentalities include "a wireless local area network (WLAN) modem to communicate data for Internet service activities between the device and at least one WLAN, when configured for and connected to the WLAN." For example, Google's devices, including the Google Pixel 7 Pro, comprise a wi-fi modem for communicating over a wi-fi networks. *See, e.g.*, https://support.google.com/pixelphone/answer/7158570?visit_id=638458852688642944-1705659042&p=specs&rd=1 ; https://www.t-mobile.com/cell-phone/google-pixel-7:

Operating system	Android 13

Display	• Full-screen 6.3-inch (160.5 mm)[1] display
	• 20:9 aspect ratio
	• FHD+ (1080 x 2400) OLED at 416 PPI
	• Smooth Display (up to 90 HZ[2])
	• Always-on display
	  • At a Glance
	  • Now Playing

Dimensions and Weight[3]	**Dimensions** • 6.1 height x 2.9 width x 0.3 depth (inches) • 155.6 height x 73.2 width x 8.7 depth (mm) **Weight** • 197 g • 6.9 oz
Battery	• Beyond 24-hour battery life[4] • Up to 72-hour battery life with Extreme Battery Saver[4] • Minimum 4270 mAh • Typical 4355 mAh[5]
Memory and Storage	**Memory** • 8 GB LPDDR5 RAM **Storage** • 128 GB / 256 GB UFS 3.1 storage[9]
Processors	• Google Tensor G2 • Titan M2™ security coprocessor

Claim	Public Documentation

Additional spec details

Battery Description	4355 mAh
Ports	USB Type-C
Connectivity	Wi-Fi 6E (802.11ax) with 2.4GHz+5GHz+6GHz, HE160,MIMO, Bluetooth 5.2, NFC
Processor	Google Tensor G2
Operating System	Android
Ram	8 GB
Maximum Expandable Memory	0 GB
Wireless Network Technology Generations	4G LTE, 5G
Supported Email Platforms	GMail, Apple Mail, POP3, IMAP4, SMTP, Microsoft® Exchange, AOL, AIM, Yahoo!® Mail
Hearing Aid Compatibility	M3, T3
WEA Capable	true
Mobile Hotspot Capable	true
Frequency	GSM: 850 MHz, 900 MHz, 1800 MHz, 1900 MHz; CDMA: BC0, BC1, BC10; LTE: 1, 2, 3, 4, 5, 7, 8, 12, 13, 14, 17, 18, 19, 20, 25, 26, 28, 29, 30, 32, 38, 39, 40, 41, 42, 46, 48, 66, 71; UMTS: Band I (2100), Band II (1900), Band IV (1700/2100), Band V (850), Band VIII (900); 5G: n1, n2, n3, n5, n7, n8, n12, n14, n20, n25, n28, n30, n38, n40, n41, n48, n66, n71, n75, n76, n77, n78
Weight	6.9 Ounces
Length	0.3
Height	6.1
Width	2.9

What's in the box

- Google Pixel 7 Device
- 1m USB-C to USB-C cable (USB 2.0)
- Quick Switch Adapter
- SIM Tool

For WEA capability, see T-Mobile WEA

California residents: see the California Proposition 65 **WARNING**

Claim	Public Documentation
[1c] a non-transient memory to store	The Accused Instrumentalities include "a non-transient memory to store." For further example, Google's devices, including the Google Pixel 7 Pro, are sold with memory. As a specific example, Google Pixel 7 Pro is sold or used by T-Mobile and includes 8GB of RAM and 128GB or 256GB of storage. *See, e.g.*, https://support.google.com/pixelphone/answer/7158570?visit_id=638458852688642944-1705659042&p=specs&rd=1 ; https://www.t-mobile.com/cell-phone/google-pixel-7:

Operating system	Android 13

Display	• Full-screen 6.3-inch (160.5 mm)[1] display
	• 20:9 aspect ratio
	• FHD+ (1080 x 2400) OLED at 416 PPI
	• Smooth Display (up to 90 HZ[2])
	• Always-on display
	  • At a Glance
	  • Now Playing

Dimensions and Weight[3]	**Dimensions** • 6.1 height x 2.9 width x 0.3 depth (inches) • 155.6 height x 73.2 width x 8.7 depth (mm) **Weight** • 197 g • 6.9 oz
Battery	• Beyond 24-hour battery life[4] • Up to 72-hour battery life with Extreme Battery Saver[4] • Minimum 4270 mAh • Typical 4355 mAh[5]
Memory and Storage	**Memory** • 8 GB LPDDR5 RAM **Storage** • 128 GB / 256 GB UFS 3.1 storage[9]
Processors	• Google Tensor G2 • Titan M2™ security coprocessor

Claim	Public Documentation

Additional spec details

Battery Description	4355 mAh
Ports	USB Type-C
Connectivity	Wi-Fi 6E (802.11ax) with 2.4GHz+5GHz+6GHz, HE160,MIMO, Bluetooth 5.2, NFC
Processor	Google Tensor G2
Operating System	Android
Ram	8 GB
Maximum Expandable Memory	0 GB
Wireless Network Technology Generations	4G LTE, 5G
Supported Email Platforms	GMail, Apple Mail, POP3, IMAP4, SMTP, Microsoft® Exchange, AOL, AIM, Yahoo!® Mail
Hearing Aid Compatibility	M3, T3
WEA Capable	true
Mobile Hotspot Capable	true
Frequency	GSM: 850 MHz, 900 MHz, 1800 MHz, 1900 MHz; CDMA: BC0, BC1, BC10; LTE: 1, 2, 3, 4, 5, 7, 8, 12, 13, 14, 17, 18, 19, 20, 25, 26, 28, 29, 30, 32, 38, 39, 40, 41, 42, 46, 48, 66, 71; UMTS: Band I (2100), Band II (1900), Band IV (1700/2100), Band V (850), Band VIII (900); 5G: n1, n2, n3, n5, n7, n8, n12, n14, n20, n25, n28, n30, n38, n40, n41, n48, n66, n71, n75, n76, n77, n78
Weight	6.9 Ounces
Length	0.3
Height	6.1
Width	2.9

What's in the box

- Google Pixel 7 Device
- 1m USB-C to USB-C cable (USB 2.0)
- Quick Switch Adapter
- SIM Tool

For WEA capability, see T-Mobile WEA

California residents: see the California Proposition 65 **WARNING**

Claim	Public Documentation
[1d] a differential traffic control policy list distinguishing between a first one or more applications resident on the device and a second one or more applications and/or services resident on the device, and	The Accused Instrumentalities comprise "a differential traffic control policy list distinguishing between a first one or more applications resident on the device and a second one or more applications and/or services resident on the device." For example, Google's devices, including the Google Pixel 7 Pro,runs the Android Operating System, which includes features such as "Data Saver," "Battery Saver," "Extreme Battery Saver," "Doze Mode," "App Standby," "Adaptive Battery," and/or "JobScheduler" which include policies which distinguish between applications and/or services. *See, e.g.*, https://support.google.com/pixelphone/answer/2819524?sjid=13223854186446774975-NC#zippy=:

Claim	Public Documentation

	# Reduce and manage mobile data usage
	
	Data usage is how much data your phone uploads or downloads using mobile data. To make sure that you're not using too much data on your data plan, you can check and change your data usage.
	
	Important: Some of these steps work only on Android 8.0 and up. Learn how to check your Android version.
	
	## Check your mobile data usage
	
	1. Open your phone's Settings app.
	2. Tap **Network and Internet** > **Internet**.
	3. Next to your operator, tap Settings ⚙.
	4. At the top you'll see how much total data you use.
	5. To see graphs and details, tap **App data usage**.
	   • To pick a time period, tap the down arrow ⌄.
	   • To see how much data each app uses, look below the graph.
	
	https://support.google.com/pixelphone/answer/7055392?hl=en-AU&sjid=13223854186446774975-NC:

Claim	Public Documentation

	# Use less mobile data with Data Saver
	
	To help use less mobile data on a limited data plan, you can turn on Data Saver. This mode lets most apps and services get background data only via Wi-Fi. Currently active apps and services can use mobile data.
	
	Important: Some of these steps work only on Android 8.0 and up. Learn how to check your Android version.
	
	## Turn Data Saver on or off
	
	1. Open your phone's Settings app.
	2. Tap **Network and Internet** › Data Saver.
	3. Turn **Data Saver** on or off.
	   • In your status bar, when Data Saver is on, you'll see the Data Saver icon .
	   • You'll also see a notification at the top of your phone's Settings app.
	
	Tip: You can add Data Saver to your settings bar. Learn how to customise Quick Settings.
	
	; https://support.google.com/pixelphone/answer/6187458?hl=en&sjid=13223854186446774975-NC:

Use Battery Saver on a Pixel phone

You can set Battery Saver to turn on automatically when your phone's battery gets low. You can also turn on Battery Saver at any time. To save even more power on your Pixel 3 or later phone, including Fold, you can turn on Extreme Battery Saver.

Important: While Battery Saver is on, a Pixel phone with 5G uses 4G service. Learn what changes while Battery Saver is on.

Important: Some of these steps work only on Android 11 and up. Learn how to check your Android version.

Turn Battery Saver on or off

When Battery Saver is on, at the top of your screen, you'll find Battery Saver on . To turn Battery Saver on or off:

1. At the top of your screen, swipe down.
2. Tap Battery Saver .

Turn Battery Saver on automatically

1. Open your phone's Settings app.
2. Tap **Battery** > **Battery Saver** > **Schedule and reminders.**
3. Make sure that "Turn on based on battery level" is on.
4. Slide the percentage bar to the level you want.

Tip: The next time your battery level falls to the percentage you have set, Battery Saver will turn on automatically for you.

Claim	Public Documentation
	; https://support.google.com/pixelphone/answer/7015477?hl=en&sjid=13223854186446774975-NC:

Keep Adaptive Battery & battery optimization on

To have apps use your phone's battery only when you need them to, keep Adaptive Battery and battery optimization on. These settings are on by default.

Important: Some of these steps work only on Android 9.0 and up. Learn how to check your Android version.

The Pixel battery continuously learns your behaviors and optimizes itself based on your most recent app usage. When you set up a new device or after a factory reset, optimization may take up a few weeks to take full effect. For best results keep Adaptive Battery and Battery Optimization on .

It's normal for your Pixel battery to drain a little more than usual after a software update. This is because the phone is working hard to download and optimize the new software and get everything up and running.

If you still experience unusual battery drain after a few days, please let us know and we'll be happy to help.

Check that Adaptive Battery is on for your phone

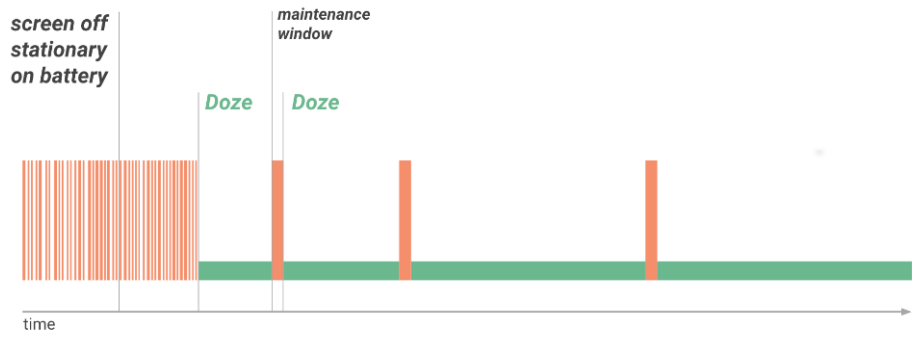
If you keep Adaptive Battery on, apps that you use less often will run less while you're not using them. Your phone can learn how you use your apps over time. This can help save battery in ways that work best for you.

1. Open your phone's Settings app.
2. Tap **Battery** > **Battery Saver**.
3. Tap **Adaptive Battery** > Expand More .
4. Turn on **Use Adaptive Battery** if it's turned off.

Tip: Adaptive Battery learns from your phone usage to continuously optimize how apps use battery. To extend battery life, it may reduce performance and delay notifications.

Claim	Public Documentation

	; https://developer.android.com/training/basics/network-ops/data-saver:
	
	Optimize network data usage
	
	Over the life of a smartphone, the cost of a cellular data plan can easily exceed the cost of the device itself. On Android 7.0 (API level 24) and higher, users can enable Data Saver on a device-wide basis in order to optimize their device's data usage, and use less data. This ability is especially useful when roaming, near the end of the billing cycle, or for a small prepaid data pack.
	
	When a user enables Data Saver in **Settings** and the device is on a metered network, the system blocks background data usage and signals apps to use less data in the foreground wherever possible. Users can allow specific apps to use background metered data usage even when Data Saver is turned on.
	
	Android 7.0 (API level 24) extends the `ConnectivityManager` API to provide apps with a way to retrieve the user's Data Saver preferences and monitor preference changes. It is considered good practice for apps to check whether the user has enabled Data Saver and make an effort to limit foreground and background data usage.
	
	Check data saver preferences
	
	On Android 7.0 (API level 24) and higher, apps can use the `ConnectivityManager` API to determine what data usage restrictions are being applied. The `getRestrictBackgroundStatus()` method returns one of the following values:
	
	`RESTRICT_BACKGROUND_STATUS_DISABLED`
	
	Data Saver is disabled.
	
	`RESTRICT_BACKGROUND_STATUS_ENABLED`
	
	The user has enabled Data Saver for this app. Apps should make an effort to limit data usage in the foreground and gracefully handle restrictions to background data usage.
	
	`RESTRICT_BACKGROUND_STATUS_WHITELISTED`
	
	The user has enabled Data Saver but the app is allowed to bypass it. Apps should still make an effort to limit foreground and background data usage.
	
	Limit data usage whenever the device is connected to a metered network, even if Data Saver is disabled or the app is allowed to bypass it. The following sample code uses `ConnectivityManager.isActiveNetworkMetered()` and `ConnectivityManager.getRestrictBackgroundStatus()` to determine how much data the app should use:
	
	; https://developer.android.com/training/monitoring-device-state/doze-standby:

Claim	Public Documentation
	Optimize for Doze and App Standby Starting from Android 6.0 (API level 23), Android introduces two power-saving features that extend battery life for users by managing how apps behave when a device is not connected to a power source. *Doze* reduces battery consumption by deferring background CPU and network activity for apps when the device is unused for long periods of time. *App Standby* defers background network activity for apps with which the user has not recently interacted. While the device is in Doze, apps' access to certain battery-intensive resources is deferred until maintenance windows. The specific restrictions are listed in Power Management Restrictions. Doze and App Standby manage the behavior of all apps running on Android 6.0 or higher, regardless whether they are specifically targeting API level 23. To ensure the best experience for users, test your app in Doze and App Standby modes and make any necessary adjustments to your code. The sections below provide details. **Understanding Doze** If a user leaves a device unplugged and stationary for a period of time, with the screen off, the device enters Doze mode. In Doze mode, the system attempts to conserve battery by restricting apps' access to network and CPU-intensive services. It also prevents apps from accessing the network and defers their jobs, syncs, and standard alarms. Periodically, the system exits Doze for a brief time to let apps complete their deferred activities. During this *maintenance window*, the system runs all pending syncs, jobs, and alarms, and lets apps access the network.  **Figure 1.** Doze provides a recurring maintenance window for apps to use the network and handle pending activities.

Claim	Public Documentation
	At the conclusion of each maintenance window, the system again enters Doze, suspending network access and deferring jobs, syncs, and alarms. Over time, the system schedules maintenance windows less and less frequently, helping to reduce battery consumption in cases of longer-term inactivity when the device is not connected to a charger. As soon as the user wakes the device by moving it, turning on the screen, or connecting a charger, the system exits Doze and all apps return to normal activity. The Doze restriction on network access is also likely to affect your app, especially if the app relies on real-time messages such as tickles or notifications. If your app requires a persistent connection to the network to receive messages, you should use Firebase Cloud Messaging (FCM) if possible. ; https://developer.android.com/topic/performance/appstandby:

App Standby Buckets

Android 9 (API level 28) and higher support **App Standby Buckets**. App Standby Buckets help the system prioritize apps' requests for resources based on how recently and how frequently the apps are used. Based on app usage patterns, each app is placed in one of five priority **buckets**. The system limits the device resources available to each app based on which bucket the app is in.

Priority buckets

The system dynamically assigns each app to a priority bucket, reassigning the apps as needed. The system may rely on a preloaded app that uses machine learning to determine how likely each app is to be used, and assigns apps to the appropriate buckets. If the system app is not present on a device, the system defaults to sorting apps based on how recently they were used. More active apps are assigned to buckets that give the apps higher priority, making more system resources available to the app. In particular, the bucket determines how frequently the app's jobs run, and how often the app can trigger alarms. These restrictions apply only while the device is on battery power; the system does not impose these restrictions on apps while the device is charging.

Note: Every manufacturer can set their own criteria for how non-active apps are assigned to buckets. You should not try to influence which bucket your app is assigned to. Instead, focus on making sure your app behaves well in whatever bucket it might be in. Your app can find out what bucket it's currently in by calling `UsageStatsManager.getAppStandbyBucket()`.

The buckets are:

1. Active: App is currently being used or was very recently used.
2. Working set: App is in regular use.
3. Frequent: App is often used, but not every day.
4. Rare: App is not frequently used.
5. Restricted: App consumes a great deal of system resources, or may exhibit undesirable behavior.

In addition, there's a special **never** bucket for apps that have been installed but have never been run. The system imposes severe restrictions on these apps.

Claim	Public Documentation
	; https://developer.android.com/topic/performance/background-optimization; https://developer.android.com/reference/android/app/job/JobScheduler; https://developer.android.com/guide/background/persistent; https://developer.android.com/guide/components/services; https://developer.android.com/guide/components/activities/intro-activities; https://developer.android.com/reference/java/net/URLConnection; https://developer.android.com/training/articles/security-ssl; https://developer.android.com/reference/android/net/DnsResolver; https://developer.android.com/guide/topics/media; https://developer.android.com/media; https://developer.android.com/guide/topics/media/platform/mediaplayer; *see also* https://techshift.net/does-data-saver-apply-to-wi-fi/: "**Does data saver apply to Wi-Fi?** Does data saver affect WiFi? **No, it doesn't**. Data saver only restricts the apps from using mobile data. While you are on WiFi, your phone's data saver won't affect it." ; https://www.technipages.com/how-to-give-android-apps-unrestricted-data-access-data-saver-on: "The Data Saver option is only when you're not on WiFi and affects how you see your content."
[1e] a differential traffic control policy applicable to at least some Internet service activities by or on behalf of the first one or more applications;	The Accused Instrumentalities comprises "a differential traffic control policy applicable to at least some Internet service activities by or on behalf of the first one or more applications." For example, Google's devices, including the Google Pixel 7 Pro, runthe Android Operating System, which includes features such as "Data Saver," "Battery Saver," "Extreme Battery Saver," "Doze Mode," "App Standby," "Adaptive Battery," and/or "JobScheduler" which include policies which apply to at least some activities by or on behalf of applications and/or services. *See, e.g.*, https://support.google.com/pixelphone/answer/2819524?sjid=13223854186446774975-NC#zippy=:

Claim	Public Documentation

	# Reduce and manage mobile data usage
	
	Data usage is how much data your phone uploads or downloads using mobile data. To make sure that you're not using too much data on your data plan, you can check and change your data usage.
	
	Important: Some of these steps work only on Android 8.0 and up. Learn how to check your Android version.
	
	## Check your mobile data usage
	
	1. Open your phone's Settings app.
	2. Tap **Network and Internet** > **Internet**.
	3. Next to your operator, tap Settings ⚙.
	4. At the top you'll see how much total data you use.
	5. To see graphs and details, tap **App data usage**.
	   • To pick a time period, tap the down arrow ⌄.
	   • To see how much data each app uses, look below the graph.
	
	https://support.google.com/pixelphone/answer/7055392?hl=en-AU&sjid=13223854186446774975-NC:

Claim	Public Documentation

	# Use less mobile data with Data Saver
	
	To help use less mobile data on a limited data plan, you can turn on Data Saver. This mode lets most apps and services get background data only via Wi-Fi. Currently active apps and services can use mobile data.
	
	Important: Some of these steps work only on Android 8.0 and up. Learn how to check your Android version.
	
	## Turn Data Saver on or off
	
	1. Open your phone's Settings app.
	2. Tap **Network and Internet** › Data Saver.
	3. Turn **Data Saver** on or off.
	   • In your status bar, when Data Saver is on, you'll see the Data Saver icon .
	   • You'll also see a notification at the top of your phone's Settings app.
	
	Tip: You can add Data Saver to your settings bar. Learn how to customise Quick Settings.
	
	; https://support.google.com/pixelphone/answer/6187458?hl=en&sjid=13223854186446774975-NC:

Use Battery Saver on a Pixel phone

You can set Battery Saver to turn on automatically when your phone's battery gets low. You can also turn on Battery Saver at any time. To save even more power on your Pixel 3 or later phone, including Fold, you can turn on Extreme Battery Saver.

Important: While Battery Saver is on, a Pixel phone with 5G uses 4G service. Learn what changes while Battery Saver is on.

Important: Some of these steps work only on Android 11 and up. Learn how to check your Android version.

Turn Battery Saver on or off

When Battery Saver is on, at the top of your screen, you'll find Battery Saver on . To turn Battery Saver on or off:

1. At the top of your screen, swipe down.
2. Tap Battery Saver .

Turn Battery Saver on automatically

1. Open your phone's Settings app.
2. Tap **Battery** > **Battery Saver** > **Schedule and reminders.**
3. Make sure that "Turn on based on battery level" is on.
4. Slide the percentage bar to the level you want.

Tip: The next time your battery level falls to the percentage you have set, Battery Saver will turn on automatically for you.

Claim	Public Documentation
	; https://support.google.com/pixelphone/answer/7015477?hl=en&sjid=13223854186446774975-NC:

Keep Adaptive Battery & battery optimization on

To have apps use your phone's battery only when you need them to, keep Adaptive Battery and battery optimization on. These settings are on by default.

Important: Some of these steps work only on Android 9.0 and up. Learn how to check your Android version.

The Pixel battery continuously learns your behaviors and optimizes itself based on your most recent app usage. When you set up a new device or after a factory reset, optimization may take up a few weeks to take full effect. For best results keep Adaptive Battery and Battery Optimization on .

It's normal for your Pixel battery to drain a little more than usual after a software update. This is because the phone is working hard to download and optimize the new software and get everything up and running.

If you still experience unusual battery drain after a few days, please let us know and we'll be happy to help.

Check that Adaptive Battery is on for your phone

If you keep Adaptive Battery on, apps that you use less often will run less while you're not using them. Your phone can learn how you use your apps over time. This can help save battery in ways that work best for you.

1. Open your phone's Settings app.
2. Tap **Battery** > **Battery Saver**.
3. Tap **Adaptive Battery** > Expand More .
4. Turn on **Use Adaptive Battery** if it's turned off.

Tip: Adaptive Battery learns from your phone usage to continuously optimize how apps use battery. To extend battery life, it may reduce performance and delay notifications.

Claim	Public Documentation
	; https://developer.android.com/training/basics/network-ops/data-saver:

Optimize network data usage

Over the life of a smartphone, the cost of a cellular data plan can easily exceed the cost of the device itself. On Android 7.0 (API level 24) and higher, users can enable Data Saver on a device-wide basis in order to optimize their device's data usage, and use less data. This ability is especially useful when roaming, near the end of the billing cycle, or for a small prepaid data pack.

When a user enables Data Saver in **Settings** and the device is on a metered network, the system blocks background data usage and signals apps to use less data in the foreground wherever possible. Users can allow specific apps to use background metered data usage even when Data Saver is turned on.

Android 7.0 (API level 24) extends the `ConnectivityManager` API to provide apps with a way to retrieve the user's Data Saver preferences and monitor preference changes. It is considered good practice for apps to check whether the user has enabled Data Saver and make an effort to limit foreground and background data usage.

Check data saver preferences

On Android 7.0 (API level 24) and higher, apps can use the `ConnectivityManager` API to determine what data usage restrictions are being applied. The `getRestrictBackgroundStatus()` method returns one of the following values:

`RESTRICT_BACKGROUND_STATUS_DISABLED`

Data Saver is disabled.

`RESTRICT_BACKGROUND_STATUS_ENABLED`

The user has enabled Data Saver for this app. Apps should make an effort to limit data usage in the foreground and gracefully handle restrictions to background data usage.

`RESTRICT_BACKGROUND_STATUS_WHITELISTED`

The user has enabled Data Saver but the app is allowed to bypass it. Apps should still make an effort to limit foreground and background data usage.

Limit data usage whenever the device is connected to a metered network, even if Data Saver is disabled or the app is allowed to bypass it. The following sample code uses `ConnectivityManager.isActiveNetworkMetered()` and `ConnectivityManager.getRestrictBackgroundStatus()` to determine how much data the app should use:

Claim	Public Documentation
	; https://developer.android.com/training/monitoring-device-state/doze-standby:

Optimize for Doze and App Standby

Starting from Android 6.0 (API level 23), Android introduces two power-saving features that extend battery life for users by managing how apps behave when a device is not connected to a power source. *Doze* reduces battery consumption by deferring background CPU and network activity for apps when the device is unused for long periods of time. *App Standby* defers background network activity for apps with which the user has not recently interacted.

While the device is in Doze, apps' access to certain battery-intensive resources is deferred until maintenance windows. The specific restrictions are listed in Power Management Restrictions.

Doze and App Standby manage the behavior of all apps running on Android 6.0 or higher, regardless whether they are specifically targeting API level 23. To ensure the best experience for users, test your app in Doze and App Standby modes and make any necessary adjustments to your code. The sections below provide details.

Understanding Doze

If a user leaves a device unplugged and stationary for a period of time, with the screen off, the device enters Doze mode. In Doze mode, the system attempts to conserve battery by restricting apps' access to network and CPU-intensive services. It also prevents apps from accessing the network and defers their jobs, syncs, and standard alarms.

Periodically, the system exits Doze for a brief time to let apps complete their deferred activities. During this *maintenance window*, the system runs all pending syncs, jobs, and alarms, and lets apps access the network.

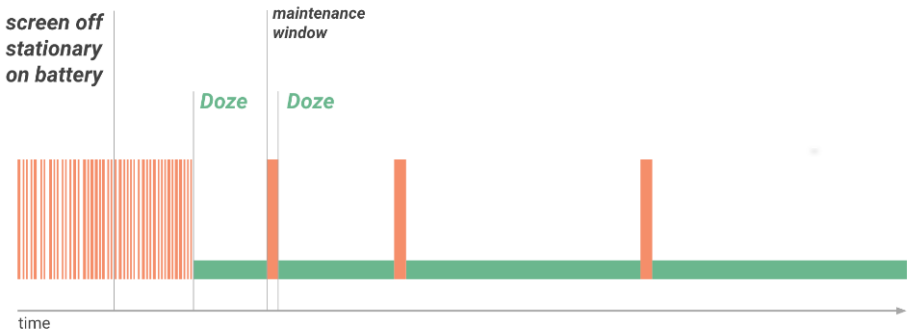


Figure 1. Doze provides a recurring maintenance window for apps to use the network and handle pending activities.

Claim	Public Documentation
	At the conclusion of each maintenance window, the system again enters Doze, suspending network access and deferring jobs, syncs, and alarms. Over time, the system schedules maintenance windows less and less frequently, helping to reduce battery consumption in cases of longer-term inactivity when the device is not connected to a charger. As soon as the user wakes the device by moving it, turning on the screen, or connecting a charger, the system exits Doze and all apps return to normal activity. The Doze restriction on network access is also likely to affect your app, especially if the app relies on real-time messages such as tickles or notifications. If your app requires a persistent connection to the network to receive messages, you should use Firebase Cloud Messaging (FCM) if possible. ; https://developer.android.com/topic/performance/appstandby:

App Standby Buckets

Android 9 (API level 28) and higher support **App Standby Buckets**. App Standby Buckets help the system prioritize apps' requests for resources based on how recently and how frequently the apps are used. Based on app usage patterns, each app is placed in one of five priority **buckets**. The system limits the device resources available to each app based on which bucket the app is in.

Priority buckets

The system dynamically assigns each app to a priority bucket, reassigning the apps as needed. The system may rely on a preloaded app that uses machine learning to determine how likely each app is to be used, and assigns apps to the appropriate buckets. If the system app is not present on a device, the system defaults to sorting apps based on how recently they were used. More active apps are assigned to buckets that give the apps higher priority, making more system resources available to the app. In particular, the bucket determines how frequently the app's jobs run, and how often the app can trigger alarms. These restrictions apply only while the device is on battery power; the system does not impose these restrictions on apps while the device is charging.

Note: Every manufacturer can set their own criteria for how non-active apps are assigned to buckets. You should not try to influence which bucket your app is assigned to. Instead, focus on making sure your app behaves well in whatever bucket it might be in. Your app can find out what bucket it's currently in by calling `UsageStatsManager.getAppStandbyBucket()`.

The buckets are:

1. Active: App is currently being used or was very recently used.
2. Working set: App is in regular use.
3. Frequent: App is often used, but not every day.
4. Rare: App is not frequently used.
5. Restricted: App consumes a great deal of system resources, or may exhibit undesirable behavior.

In addition, there's a special **never** bucket for apps that have been installed but have never been run. The system imposes severe restrictions on these apps.

Claim	Public Documentation
	; https://developer.android.com/topic/performance/background-optimization; https://developer.android.com/reference/android/app/job/JobScheduler; https://developer.android.com/guide/background/persistent; https://developer.android.com/guide/components/services; https://developer.android.com/guide/components/activities/intro-activities; https://developer.android.com/reference/java/net/URLConnection; https://developer.android.com/training/articles/security-ssl; https://developer.android.com/reference/android/net/DnsResolver; https://developer.android.com/guide/topics/media; https://developer.android.com/media; https://developer.android.com/guide/topics/media/platform/mediaplayer; *see also* https://techshift.net/does-data-saver-apply-to-wi-fi/: "**Does data saver apply to Wi-Fi?** Does data saver affect WiFi? **No, it doesn't**. Data saver only restricts the apps from using mobile data. While you are on WiFi, your phone's data saver won't affect it." ; https://www.technipages.com/how-to-give-android-apps-unrestricted-data-access-data-saver-on: "The Data Saver option is only when you're not on WiFi and affects how you see your content."
[1f] an interface to allow a user to augment the differential traffic control policy for the first one or more applications but not for the second one or more applications and/or services; and	The Accused Instrumentalities include "an interface to allow a user to augment the differential traffic control policy for the first one or more applications but not for the second one or more applications and/or services." For example, Google's devices, including the Google Pixel 7 Pro, include an interface which allow users to augment policies and settings for some applications and/or services, but not all applications and/or services (*e.g.*, system services). *See, e.g.*, https://support.google.com/pixelphone/answer/2819524?sjid=13223854186446774975-NC#zippy=:

Claim	Public Documentation

	# Reduce and manage mobile data usage
	
	Data usage is how much data your phone uploads or downloads using mobile data. To make sure that you're not using too much data on your data plan, you can check and change your data usage.
	
	Important: Some of these steps work only on Android 8.0 and up. Learn how to check your Android version.
	
	## Check your mobile data usage
	
	1. Open your phone's Settings app.
	2. Tap **Network and Internet** > **Internet**.
	3. Next to your operator, tap Settings ⚙.
	4. At the top you'll see how much total data you use.
	5. To see graphs and details, tap **App data usage**.
	   • To pick a time period, tap the down arrow ⌄.
	   • To see how much data each app uses, look below the graph.
	
	https://support.google.com/pixelphone/answer/7055392?hl=en-AU&sjid=13223854186446774975-NC:

Claim	Public Documentation

	# Use less mobile data with Data Saver
	
	To help use less mobile data on a limited data plan, you can turn on Data Saver. This mode lets most apps and services get background data only via Wi-Fi. Currently active apps and services can use mobile data.
	
	Important: Some of these steps work only on Android 8.0 and up. Learn how to check your Android version.
	
	## Turn Data Saver on or off
	
	1. Open your phone's Settings app.
	2. Tap **Network and Internet** › Data Saver.
	3. Turn **Data Saver** on or off.
	   • In your status bar, when Data Saver is on, you'll see the Data Saver icon .
	   • You'll also see a notification at the top of your phone's Settings app.
	
	Tip: You can add Data Saver to your settings bar. Learn how to customise Quick Settings.
	
	; https://support.google.com/pixelphone/answer/6187458?hl=en&sjid=13223854186446774975-NC:

Use Battery Saver on a Pixel phone

You can set Battery Saver to turn on automatically when your phone's battery gets low. You can also turn on Battery Saver at any time. To save even more power on your Pixel 3 or later phone, including Fold, you can turn on Extreme Battery Saver.

Important: While Battery Saver is on, a Pixel phone with 5G uses 4G service. Learn what changes while Battery Saver is on.

Important: Some of these steps work only on Android 11 and up. Learn how to check your Android version.

Turn Battery Saver on or off

When Battery Saver is on, at the top of your screen, you'll find Battery Saver on . To turn Battery Saver on or off:

1. At the top of your screen, swipe down.
2. Tap Battery Saver .

Turn Battery Saver on automatically

1. Open your phone's Settings app.
2. Tap **Battery** > **Battery Saver** > **Schedule and reminders.**
3. Make sure that "Turn on based on battery level" is on.
4. Slide the percentage bar to the level you want.

Tip: The next time your battery level falls to the percentage you have set, Battery Saver will turn on automatically for you.

Claim	Public Documentation
	; https://support.google.com/pixelphone/answer/7015477?hl=en&sjid=13223854186446774975-NC:

Keep Adaptive Battery & battery optimization on

To have apps use your phone's battery only when you need them to, keep Adaptive Battery and battery optimization on. These settings are on by default.

Important: Some of these steps work only on Android 9.0 and up. Learn how to check your Android version.

The Pixel battery continuously learns your behaviors and optimizes itself based on your most recent app usage. When you set up a new device or after a factory reset, optimization may take up a few weeks to take full effect. For best results keep Adaptive Battery and Battery Optimization on .

It's normal for your Pixel battery to drain a little more than usual after a software update. This is because the phone is working hard to download and optimize the new software and get everything up and running.

If you still experience unusual battery drain after a few days, please let us know and we'll be happy to help.

Check that Adaptive Battery is on for your phone

If you keep Adaptive Battery on, apps that you use less often will run less while you're not using them. Your phone can learn how you use your apps over time. This can help save battery in ways that work best for you.

1. Open your phone's Settings app.
2. Tap **Battery** > **Battery Saver**.
3. Tap **Adaptive Battery** > Expand More .
4. Turn on **Use Adaptive Battery** if it's turned off.

Tip: Adaptive Battery learns from your phone usage to continuously optimize how apps use battery. To extend battery life, it may reduce performance and delay notifications.

Claim	Public Documentation
	; https://developer.android.com/training/basics/network-ops/data-saver:

Optimize network data usage

Over the life of a smartphone, the cost of a cellular data plan can easily exceed the cost of the device itself. On Android 7.0 (API level 24) and higher, users can enable Data Saver on a device-wide basis in order to optimize their device's data usage, and use less data. This ability is especially useful when roaming, near the end of the billing cycle, or for a small prepaid data pack.

When a user enables Data Saver in **Settings** and the device is on a metered network, the system blocks background data usage and signals apps to use less data in the foreground wherever possible. Users can allow specific apps to use background metered data usage even when Data Saver is turned on.

Android 7.0 (API level 24) extends the `ConnectivityManager` API to provide apps with a way to retrieve the user's Data Saver preferences and monitor preference changes. It is considered good practice for apps to check whether the user has enabled Data Saver and make an effort to limit foreground and background data usage.

Check data saver preferences

On Android 7.0 (API level 24) and higher, apps can use the `ConnectivityManager` API to determine what data usage restrictions are being applied. The `getRestrictBackgroundStatus()` method returns one of the following values:

`RESTRICT_BACKGROUND_STATUS_DISABLED`

Data Saver is disabled.

`RESTRICT_BACKGROUND_STATUS_ENABLED`

The user has enabled Data Saver for this app. Apps should make an effort to limit data usage in the foreground and gracefully handle restrictions to background data usage.

`RESTRICT_BACKGROUND_STATUS_WHITELISTED`

The user has enabled Data Saver but the app is allowed to bypass it. Apps should still make an effort to limit foreground and background data usage.

Limit data usage whenever the device is connected to a metered network, even if Data Saver is disabled or the app is allowed to bypass it. The following sample code uses `ConnectivityManager.isActiveNetworkMetered()` and `ConnectivityManager.getRestrictBackgroundStatus()` to determine how much data the app should use:

Claim	Public Documentation
	; https://developer.android.com/training/monitoring-device-state/doze-standby:

Optimize for Doze and App Standby

Starting from Android 6.0 (API level 23), Android introduces two power-saving features that extend battery life for users by managing how apps behave when a device is not connected to a power source. *Doze* reduces battery consumption by deferring background CPU and network activity for apps when the device is unused for long periods of time. *App Standby* defers background network activity for apps with which the user has not recently interacted.

While the device is in Doze, apps' access to certain battery-intensive resources is deferred until maintenance windows. The specific restrictions are listed in Power Management Restrictions.

Doze and App Standby manage the behavior of all apps running on Android 6.0 or higher, regardless whether they are specifically targeting API level 23. To ensure the best experience for users, test your app in Doze and App Standby modes and make any necessary adjustments to your code. The sections below provide details.

Understanding Doze

If a user leaves a device unplugged and stationary for a period of time, with the screen off, the device enters Doze mode. In Doze mode, the system attempts to conserve battery by restricting apps' access to network and CPU-intensive services. It also prevents apps from accessing the network and defers their jobs, syncs, and standard alarms.

Periodically, the system exits Doze for a brief time to let apps complete their deferred activities. During this *maintenance window*, the system runs all pending syncs, jobs, and alarms, and lets apps access the network.

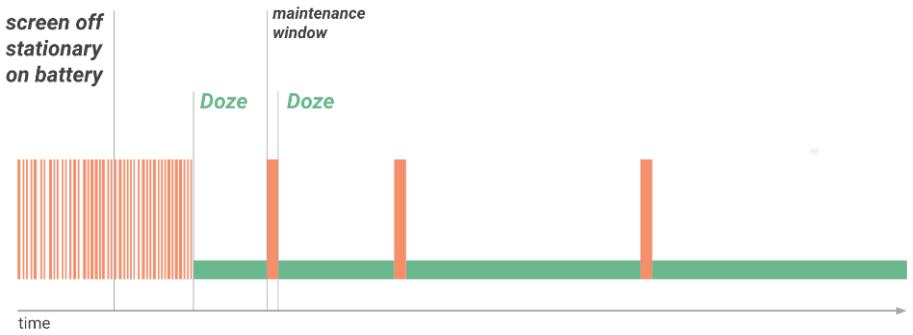


Figure 1. Doze provides a recurring maintenance window for apps to use the network and handle pending activities.

Claim	Public Documentation
	At the conclusion of each maintenance window, the system again enters Doze, suspending network access and deferring jobs, syncs, and alarms. Over time, the system schedules maintenance windows less and less frequently, helping to reduce battery consumption in cases of longer-term inactivity when the device is not connected to a charger. As soon as the user wakes the device by moving it, turning on the screen, or connecting a charger, the system exits Doze and all apps return to normal activity. The Doze restriction on network access is also likely to affect your app, especially if the app relies on real-time messages such as tickles or notifications. If your app requires a persistent connection to the network to receive messages, you should use Firebase Cloud Messaging (FCM) if possible. ; https://developer.android.com/topic/performance/appstandby:

App Standby Buckets

Android 9 (API level 28) and higher support **App Standby Buckets**. App Standby Buckets help the system prioritize apps' requests for resources based on how recently and how frequently the apps are used. Based on app usage patterns, each app is placed in one of five priority **buckets**. The system limits the device resources available to each app based on which bucket the app is in.

Priority buckets

The system dynamically assigns each app to a priority bucket, reassigning the apps as needed. The system may rely on a preloaded app that uses machine learning to determine how likely each app is to be used, and assigns apps to the appropriate buckets. If the system app is not present on a device, the system defaults to sorting apps based on how recently they were used. More active apps are assigned to buckets that give the apps higher priority, making more system resources available to the app. In particular, the bucket determines how frequently the app's jobs run, and how often the app can trigger alarms. These restrictions apply only while the device is on battery power; the system does not impose these restrictions on apps while the device is charging.

Note: Every manufacturer can set their own criteria for how non-active apps are assigned to buckets. You should not try to influence which bucket your app is assigned to. Instead, focus on making sure your app behaves well in whatever bucket it might be in. Your app can find out what bucket it's currently in by calling `UsageStatsManager.getAppStandbyBucket()`.

The buckets are:

1. Active: App is currently being used or was very recently used.
2. Working set: App is in regular use.
3. Frequent: App is often used, but not every day.
4. Rare: App is not frequently used.
5. Restricted: App consumes a great deal of system resources, or may exhibit undesirable behavior.

In addition, there's a special **never** bucket for apps that have been installed but have never been run. The system imposes severe restrictions on these apps.

Claim	Public Documentation
	; https://developer.android.com/topic/performance/background-optimization; https://developer.android.com/reference/android/app/job/JobScheduler; https://developer.android.com/guide/background/persistent; https://developer.android.com/guide/components/services; https://developer.android.com/guide/components/activities/intro-activities; https://developer.android.com/reference/java/net/URLConnection; https://developer.android.com/training/articles/security-ssl; https://developer.android.com/reference/android/net/DnsResolver; https://developer.android.com/guide/topics/media; https://developer.android.com/media; https://developer.android.com/guide/topics/media/platform/mediaplayer.
[1g] one or more processors configured to	The Accused Instrumentalities include "one or more processors." For example, Google's devices, including the Google Pixel 7 Pro, comprise one or more processors. As a specific example, the Google Pixel 7 Pro sold or used by T-Mobile include at least a Google Tensor G2 and Titan M2 Security Processor. *See, e.g.*, https://support.google.com/pixelphone/answer/7158570?visit_id=638458852688642944-1705659042&p=specs&rd=1; https://www.t-mobile.com/cell-phone/google-pixel-7:

Operating system	Android 13

Display	• Full-screen 6.3-inch (160.5 mm)[1] display
	• 20:9 aspect ratio
	• FHD+ (1080 x 2400) OLED at 416 PPI
	• Smooth Display (up to 90 HZ[2])
	• Always-on display
	  • At a Glance
	  • Now Playing

Dimensions and Weight[3]	**Dimensions** • 6.1 height x 2.9 width x 0.3 depth (inches) • 155.6 height x 73.2 width x 8.7 depth (mm) **Weight** • 197 g • 6.9 oz
Battery	• Beyond 24-hour battery life[4] • Up to 72-hour battery life with Extreme Battery Saver[4] • Minimum 4270 mAh • Typical 4355 mAh[5]
Memory and Storage	**Memory** • 8 GB LPDDR5 RAM **Storage** • 128 GB / 256 GB UFS 3.1 storage[9]
Processors	• Google Tensor G2 • Titan M2™ security coprocessor

Claim	Public Documentation

Additional spec details

Battery Description	4355 mAh
Ports	USB Type-C
Connectivity	Wi-Fi 6E (802.11ax) with 2.4GHz+5GHz+6GHz, HE160,MIMO, Bluetooth 5.2, NFC
Processor	Google Tensor G2
Operating System	Android
Ram	8 GB
Maximum Expandable Memory	0 GB
Wireless Network Technology Generations	4G LTE, 5G
Supported Email Platforms	GMail, Apple Mail, POP3, IMAP4, SMTP, Microsoft® Exchange, AOL, AIM, Yahoo!® Mail
Hearing Aid Compatibility	M3, T3
WEA Capable	true
Mobile Hotspot Capable	true
Frequency	GSM: 850 MHz, 900 MHz, 1800 MHz, 1900 MHz; CDMA: BC0, BC1, BC10; LTE: 1, 2, 3, 4, 5, 7, 8, 12, 13, 14, 17, 18, 19, 20, 25, 26, 28, 29, 30, 32, 38, 39, 40, 41, 42, 46, 48, 66, 71; UMTS: Band I (2100), Band II (1900), Band IV (1700/2100), Band V (850), Band VIII (900); 5G: n1, n2, n3, n5, n7, n8, n12, n14, n20, n25, n28, n30, n38, n40, n41, n48, n66, n71, n75, n76, n77, n78
Weight	6.9 Ounces
Length	0.3
Height	6.1
Width	2.9

What's in the box

- Google Pixel 7 Device
- 1m USB-C to USB-C cable (USB 2.0)
- Quick Switch Adapter
- SIM Tool

For WEA capability, see T-Mobile WEA

California residents: see the California Proposition 65 **WARNING**

Claim	Public Documentation
[1h] classify a wireless network to which the device currently connects in order to communicate data for Internet service activities as at least one of a plurality of network types that the device can connect with,	The Accused Instrumentalities "classify a wireless network to which the device currently connects in order to communicate data for Internet service activities as at least one of a plurality of network types that the device can connect with." For example, Google's devices, including the Google Pixel 7 Pro, sold and used by T-Mobile classify wireless network connections for communicating internet service activities. *See, e.g.*, https://support.google.com/pixelphone/answer/2819524?sjid=13223854186446774975-NC#zippy=:

Claim	Public Documentation

	# Reduce and manage mobile data usage
	
	Data usage is how much data your phone uploads or downloads using mobile data. To make sure that you're not using too much data on your data plan, you can check and change your data usage.
	
	Important: Some of these steps work only on Android 8.0 and up. Learn how to check your Android version.
	
	## Check your mobile data usage
	
	1. Open your phone's Settings app.
	2. Tap **Network and Internet** > **Internet**.
	3. Next to your operator, tap Settings ⚙.
	4. At the top you'll see how much total data you use.
	5. To see graphs and details, tap **App data usage**.
	   • To pick a time period, tap the down arrow ⌄.
	   • To see how much data each app uses, look below the graph.
	
	https://support.google.com/pixelphone/answer/7055392?hl=en-AU&sjid=13223854186446774975-NC:

Claim	Public Documentation

	# Use less mobile data with Data Saver
	
	To help use less mobile data on a limited data plan, you can turn on Data Saver. This mode lets most apps and services get background data only via Wi-Fi. Currently active apps and services can use mobile data.
	
	Important: Some of these steps work only on Android 8.0 and up. Learn how to check your Android version.
	
	## Turn Data Saver on or off
	
	1. Open your phone's Settings app.
	2. Tap **Network and Internet** › Data Saver.
	3. Turn **Data Saver** on or off.
	   • In your status bar, when Data Saver is on, you'll see the Data Saver icon .
	   • You'll also see a notification at the top of your phone's Settings app.
	
	Tip: You can add Data Saver to your settings bar. Learn how to customise Quick Settings.
	
	; https://support.google.com/pixelphone/answer/6187458?hl=en&sjid=13223854186446774975-NC:

Use Battery Saver on a Pixel phone

You can set Battery Saver to turn on automatically when your phone's battery gets low. You can also turn on Battery Saver at any time. To save even more power on your Pixel 3 or later phone, including Fold, you can turn on Extreme Battery Saver.

Important: While Battery Saver is on, a Pixel phone with 5G uses 4G service. Learn what changes while Battery Saver is on.

Important: Some of these steps work only on Android 11 and up. Learn how to check your Android version.

Turn Battery Saver on or off

When Battery Saver is on, at the top of your screen, you'll find Battery Saver on . To turn Battery Saver on or off:

1. At the top of your screen, swipe down.
2. Tap Battery Saver .

Turn Battery Saver on automatically

1. Open your phone's Settings app.
2. Tap **Battery** > **Battery Saver** > **Schedule and reminders.**
3. Make sure that "Turn on based on battery level" is on.
4. Slide the percentage bar to the level you want.

Tip: The next time your battery level falls to the percentage you have set, Battery Saver will turn on automatically for you.

Claim	Public Documentation
	; https://support.google.com/pixelphone/answer/7015477?hl=en&sjid=13223854186446774975-NC:

Keep Adaptive Battery & battery optimization on

To have apps use your phone's battery only when you need them to, keep Adaptive Battery and battery optimization on. These settings are on by default.

Important: Some of these steps work only on Android 9.0 and up. Learn how to check your Android version.

The Pixel battery continuously learns your behaviors and optimizes itself based on your most recent app usage. When you set up a new device or after a factory reset, optimization may take up a few weeks to take full effect. For best results keep Adaptive Battery and Battery Optimization on .

It's normal for your Pixel battery to drain a little more than usual after a software update. This is because the phone is working hard to download and optimize the new software and get everything up and running.

If you still experience unusual battery drain after a few days, please let us know and we'll be happy to help.

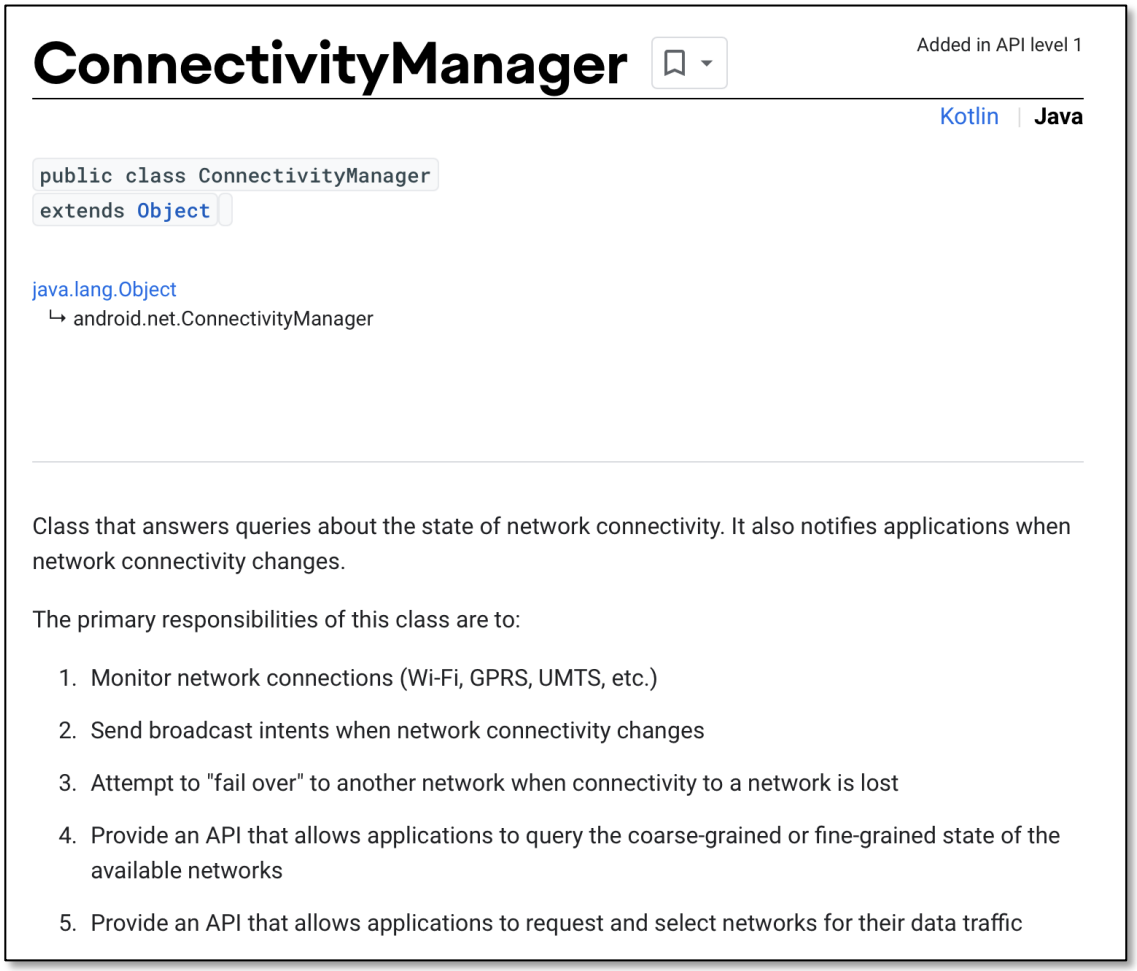
Check that Adaptive Battery is on for your phone

If you keep Adaptive Battery on, apps that you use less often will run less while you're not using them. Your phone can learn how you use your apps over time. This can help save battery in ways that work best for you.

1. Open your phone's Settings app.
2. Tap **Battery** > **Battery Saver**.
3. Tap **Adaptive Battery** > Expand More .
4. Turn on **Use Adaptive Battery** if it's turned off.

Tip: Adaptive Battery learns from your phone usage to continuously optimize how apps use battery. To extend battery life, it may reduce performance and delay notifications.

; https://developer.android.com/reference/android/net/ConnectivityManager:



ConnectivityManager

Added in API level 1

Kotlin | **Java**

```
public class ConnectivityManager
extends Object
```

java.lang.Object
↳ android.net.ConnectivityManager

Class that answers queries about the state of network connectivity. It also notifies applications when network connectivity changes.

The primary responsibilities of this class are to:

1. Monitor network connections (Wi-Fi, GPRS, UMTS, etc.)
2. Send broadcast intents when network connectivity changes
3. Attempt to "fail over" to another network when connectivity to a network is lost
4. Provide an API that allows applications to query the coarse-grained or fine-grained state of the available networks
5. Provide an API that allows applications to request and select networks for their data traffic

; https://developer.android.com/training/monitoring-device-state/connectivity-status-type; https://developer.android.com/training/basics/network-ops/data-saver; https://developer.android.com/training/monitoring-device-state/doze-standby; https://developer.android.com/topic/performance/appstandby:

App Standby Buckets

Android 9 (API level 28) and higher support **App Standby Buckets**. App Standby Buckets help the system prioritize apps' requests for resources based on how recently and how frequently the apps are used. Based on app usage patterns, each app is placed in one of five priority **buckets**. The system limits the device resources available to each app based on which bucket the app is in.

Priority buckets

The system dynamically assigns each app to a priority bucket, reassigning the apps as needed. The system may rely on a preloaded app that uses machine learning to determine how likely each app is to be used, and assigns apps to the appropriate buckets. If the system app is not present on a device, the system defaults to sorting apps based on how recently they were used. More active apps are assigned to buckets that give the apps higher priority, making more system resources available to the app. In particular, the bucket determines how frequently the app's jobs run, and how often the app can trigger alarms. These restrictions apply only while the device is on battery power; the system does not impose these restrictions on apps while the device is charging.

Note: Every manufacturer can set their own criteria for how non-active apps are assigned to buckets. You should not try to influence which bucket your app is assigned to. Instead, focus on making sure your app behaves well in whatever bucket it might be in. Your app can find out what bucket it's currently in by calling `UsageStatsManager.getAppStandbyBucket()`.

The buckets are:

1. Active: App is currently being used or was very recently used.
2. Working set: App is in regular use.
3. Frequent: App is often used, but not every day.
4. Rare: App is not frequently used.
5. Restricted: App consumes a great deal of system resources, or may exhibit undesirable behavior.

In addition, there's a special **never** bucket for apps that have been installed but have never been run. The system imposes severe restrictions on these apps.

Claim	Public Documentation
	★ **Note:** Unlike other buckets, these power management restrictions apply to the restricted bucket even when the device is charging. However, restrictions are loosened when the device is charging, idle, and on an unmetered network. ; https://developer.android.com/topic/performance/background-optimization; https://developer.android.com/reference/android/app/job/JobScheduler; https://developer.android.com/guide/background/persistent; https://developer.android.com/guide/components/services; https://developer.android.com/guide/components/activities/intro-activities; https://developer.android.com/reference/java/net/URLConnection; https://developer.android.com/training/articles/security-ssl; https://developer.android.com/reference/android/net/DnsResolver; https://developer.android.com/guide/topics/media; https://developer.android.com/media; https://developer.android.com/guide/topics/media/platform/mediaplayer; https://techshift.net/does-data-saver-apply-to-wi-fi/: "**Does data saver apply to Wi-Fi?** Does data saver affect WiFi? **No, it doesn't**. Data saver only restricts the apps from using mobile data. While you are on WiFi, your phone's data saver won't affect it." ; https://www.technipages.com/how-to-give-android-apps-unrestricted-data-access-data-saver-on: "The Data Saver option is only when you're not on WiFi and affects how you see your content."
[1i] classify whether a particular application capable of both interacting with the user in a user interface foreground of the device, and at least some Internet service activities when not interacting with the user in the device user interface foreground,	The Accused Instrumentalities "classify whether a particular application capable of both interacting with the user in a user interface foreground of the device, and at least some Internet service activities when not interacting with the user in the device user interface foreground." For example, Google's devices, including the Google Pixel 7 Pro, sold and used by T-Mobile classify applications and internet service activities in both foreground and background. *See, e.g.*, https://support.google.com/pixelphone/answer/2819524?sjid=13223854186446774975-NC#zippy=:

Claim	Public Documentation

	# Reduce and manage mobile data usage
	
	Data usage is how much data your phone uploads or downloads using mobile data. To make sure that you're not using too much data on your data plan, you can check and change your data usage.
	
	Important: Some of these steps work only on Android 8.0 and up. Learn how to check your Android version.
	
	## Check your mobile data usage
	
	1. Open your phone's Settings app.
	2. Tap **Network and Internet** > **Internet**.
	3. Next to your operator, tap Settings ⚙.
	4. At the top you'll see how much total data you use.
	5. To see graphs and details, tap **App data usage**.
	   • To pick a time period, tap the down arrow ⌄.
	   • To see how much data each app uses, look below the graph.
	
	https://support.google.com/pixelphone/answer/7055392?hl=en-AU&sjid=13223854186446774975-NC:

Claim	Public Documentation

	# Use less mobile data with Data Saver
	
	To help use less mobile data on a limited data plan, you can turn on Data Saver. This mode lets most apps and services get background data only via Wi-Fi. Currently active apps and services can use mobile data.
	
	Important: Some of these steps work only on Android 8.0 and up. Learn how to check your Android version.
	
	## Turn Data Saver on or off
	
	1. Open your phone's Settings app.
	2. Tap **Network and Internet** › Data Saver.
	3. Turn **Data Saver** on or off.
	   • In your status bar, when Data Saver is on, you'll see the Data Saver icon .
	   • You'll also see a notification at the top of your phone's Settings app.
	
	Tip: You can add Data Saver to your settings bar. Learn how to customise Quick Settings.
	
	; https://support.google.com/pixelphone/answer/6187458?hl=en&sjid=13223854186446774975-NC:

Use Battery Saver on a Pixel phone

You can set Battery Saver to turn on automatically when your phone's battery gets low. You can also turn on Battery Saver at any time. To save even more power on your Pixel 3 or later phone, including Fold, you can turn on Extreme Battery Saver.

Important: While Battery Saver is on, a Pixel phone with 5G uses 4G service. Learn what changes while Battery Saver is on.

Important: Some of these steps work only on Android 11 and up. Learn how to check your Android version.

Turn Battery Saver on or off

When Battery Saver is on, at the top of your screen, you'll find Battery Saver on . To turn Battery Saver on or off:

1. At the top of your screen, swipe down.
2. Tap Battery Saver .

Turn Battery Saver on automatically

1. Open your phone's Settings app.
2. Tap **Battery** > **Battery Saver** > **Schedule and reminders.**
3. Make sure that "Turn on based on battery level" is on.
4. Slide the percentage bar to the level you want.

Tip: The next time your battery level falls to the percentage you have set, Battery Saver will turn on automatically for you.

Claim	Public Documentation
	; https://support.google.com/pixelphone/answer/7015477?hl=en&sjid=13223854186446774975-NC:

Keep Adaptive Battery & battery optimization on

To have apps use your phone's battery only when you need them to, keep Adaptive Battery and battery optimization on. These settings are on by default.

Important: Some of these steps work only on Android 9.0 and up. Learn how to check your Android version.

The Pixel battery continuously learns your behaviors and optimizes itself based on your most recent app usage. When you set up a new device or after a factory reset, optimization may take up a few weeks to take full effect. For best results keep Adaptive Battery and Battery Optimization on .

It's normal for your Pixel battery to drain a little more than usual after a software update. This is because the phone is working hard to download and optimize the new software and get everything up and running.

If you still experience unusual battery drain after a few days, please let us know and we'll be happy to help.

Check that Adaptive Battery is on for your phone

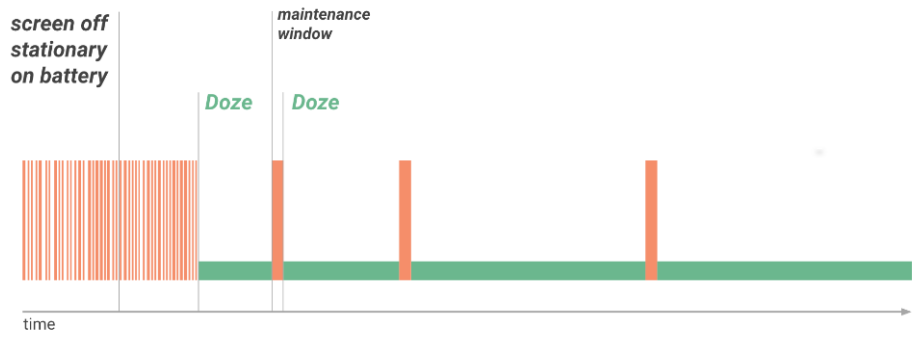
If you keep Adaptive Battery on, apps that you use less often will run less while you're not using them. Your phone can learn how you use your apps over time. This can help save battery in ways that work best for you.

1. Open your phone's Settings app.
2. Tap **Battery** > **Battery Saver**.
3. Tap **Adaptive Battery** > Expand More .
4. Turn on **Use Adaptive Battery** if it's turned off.

Tip: Adaptive Battery learns from your phone usage to continuously optimize how apps use battery. To extend battery life, it may reduce performance and delay notifications.

Claim	Public Documentation

	; https://developer.android.com/training/basics/network-ops/data-saver:
	
	Optimize network data usage
	
	Over the life of a smartphone, the cost of a cellular data plan can easily exceed the cost of the device itself. On Android 7.0 (API level 24) and higher, users can enable Data Saver on a device-wide basis in order to optimize their device's data usage, and use less data. This ability is especially useful when roaming, near the end of the billing cycle, or for a small prepaid data pack.
	
	When a user enables Data Saver in **Settings** and the device is on a metered network, the system blocks background data usage and signals apps to use less data in the foreground wherever possible. Users can allow specific apps to use background metered data usage even when Data Saver is turned on.
	
	Android 7.0 (API level 24) extends the `ConnectivityManager` API to provide apps with a way to retrieve the user's Data Saver preferences and monitor preference changes. It is considered good practice for apps to check whether the user has enabled Data Saver and make an effort to limit foreground and background data usage.
	
	Check data saver preferences
	
	On Android 7.0 (API level 24) and higher, apps can use the `ConnectivityManager` API to determine what data usage restrictions are being applied. The `getRestrictBackgroundStatus()` method returns one of the following values:
	
	`RESTRICT_BACKGROUND_STATUS_DISABLED`
	Data Saver is disabled.
	
	`RESTRICT_BACKGROUND_STATUS_ENABLED`
	The user has enabled Data Saver for this app. Apps should make an effort to limit data usage in the foreground and gracefully handle restrictions to background data usage.
	
	`RESTRICT_BACKGROUND_STATUS_WHITELISTED`
	The user has enabled Data Saver but the app is allowed to bypass it. Apps should still make an effort to limit foreground and background data usage.
	
	Limit data usage whenever the device is connected to a metered network, even if Data Saver is disabled or the app is allowed to bypass it. The following sample code uses `ConnectivityManager.isActiveNetworkMetered()` and `ConnectivityManager.getRestrictBackgroundStatus()` to determine how much data the app should use:
	
	; https://developer.android.com/training/monitoring-device-state/doze-standby:

Claim	Public Documentation
	Optimize for Doze and App Standby Starting from Android 6.0 (API level 23), Android introduces two power-saving features that extend battery life for users by managing how apps behave when a device is not connected to a power source. *Doze* reduces battery consumption by deferring background CPU and network activity for apps when the device is unused for long periods of time. *App Standby* defers background network activity for apps with which the user has not recently interacted. While the device is in Doze, apps' access to certain battery-intensive resources is deferred until maintenance windows. The specific restrictions are listed in Power Management Restrictions. Doze and App Standby manage the behavior of all apps running on Android 6.0 or higher, regardless whether they are specifically targeting API level 23. To ensure the best experience for users, test your app in Doze and App Standby modes and make any necessary adjustments to your code. The sections below provide details. **Understanding Doze** If a user leaves a device unplugged and stationary for a period of time, with the screen off, the device enters Doze mode. In Doze mode, the system attempts to conserve battery by restricting apps' access to network and CPU-intensive services. It also prevents apps from accessing the network and defers their jobs, syncs, and standard alarms. Periodically, the system exits Doze for a brief time to let apps complete their deferred activities. During this *maintenance window*, the system runs all pending syncs, jobs, and alarms, and lets apps access the network.  **Figure 1.** Doze provides a recurring maintenance window for apps to use the network and handle pending activities.

Claim	Public Documentation
	At the conclusion of each maintenance window, the system again enters Doze, suspending network access and deferring jobs, syncs, and alarms. Over time, the system schedules maintenance windows less and less frequently, helping to reduce battery consumption in cases of longer-term inactivity when the device is not connected to a charger. As soon as the user wakes the device by moving it, turning on the screen, or connecting a charger, the system exits Doze and all apps return to normal activity. The Doze restriction on network access is also likely to affect your app, especially if the app relies on real-time messages such as tickles or notifications. If your app requires a persistent connection to the network to receive messages, you should use Firebase Cloud Messaging (FCM) if possible. ; https://developer.android.com/topic/performance/appstandby:

App Standby Buckets

Android 9 (API level 28) and higher support **App Standby Buckets**. App Standby Buckets help the system prioritize apps' requests for resources based on how recently and how frequently the apps are used. Based on app usage patterns, each app is placed in one of five priority **buckets**. The system limits the device resources available to each app based on which bucket the app is in.

Priority buckets

The system dynamically assigns each app to a priority bucket, reassigning the apps as needed. The system may rely on a preloaded app that uses machine learning to determine how likely each app is to be used, and assigns apps to the appropriate buckets. If the system app is not present on a device, the system defaults to sorting apps based on how recently they were used. More active apps are assigned to buckets that give the apps higher priority, making more system resources available to the app. In particular, the bucket determines how frequently the app's jobs run, and how often the app can trigger alarms. These restrictions apply only while the device is on battery power; the system does not impose these restrictions on apps while the device is charging.

> **Note:** Every manufacturer can set their own criteria for how non-active apps are assigned to buckets. You should not try to influence which bucket your app is assigned to. Instead, focus on making sure your app behaves well in whatever bucket it might be in. Your app can find out what bucket it's currently in by calling `UsageStatsManager.getAppStandbyBucket()`.

The buckets are:

1. Active: App is currently being used or was very recently used.
2. Working set: App is in regular use.
3. Frequent: App is often used, but not every day.
4. Rare: App is not frequently used.
5. Restricted: App consumes a great deal of system resources, or may exhibit undesirable behavior.

In addition, there's a special **never** bucket for apps that have been installed but have never been run. The system imposes severe restrictions on these apps.

Claim	Public Documentation
	; https://developer.android.com/topic/performance/power/power-details; https://developer.android.com/topic/performance/background-optimization; https://developer.android.com/reference/android/app/job/JobScheduler; https://developer.android.com/guide/background/persistent; https://developer.android.com/guide/components/activities/process-lifecycle: 1. A **foreground process** is one that is required for what the user is currently doing. Various application components can cause its containing process to be considered foreground in different ways. A process is considered to be in the foreground if any of the following conditions hold: • It is running an `Activity` at the top of the screen that the user is interacting with (its `onResume()` method has been called). • It has a `BroadcastReceiver` that is currently running (its `BroadcastReceiver.onReceive()` method is executing). • It has a `Service` that is currently executing code in one of its callbacks (`Service.onCreate()`, `Service.onStart()`, or `Service.onDestroy()`). There will only ever be a few such processes in the system, and these will only be killed as a last resort if memory is so low that not even these processes can continue to run. Generally, at this point, the device has reached a memory paging state, so this action is required in order to keep the user interface responsive. ; https://developer.android.com/guide/background:

Claim	Public Documentation
	Definition of background work An app is running in the *background* when both the following conditions are satisfied: • None of the app's activities are currently visible to the user. • The app isn't running any foreground services that started while an activity from the app was visible to the user. Otherwise, the app is running in the *foreground*. ; https://developer.android.com/guide/components/services;

Types of Services

These are the three different types of services:

Foreground

A foreground service performs some operation that is noticeable to the user. For example, an audio app would use a foreground service to play an audio track. Foreground services must display a Notification. Foreground services continue running even when the user isn't interacting with the app.

When you use a foreground service, you must display a notification so that users are actively aware that the service is running. This notification cannot be dismissed unless the service is either stopped or removed from the foreground.

Learn more about how to configure foreground services in your app.

> ★ **Note:** The WorkManager API offers a flexible way of scheduling tasks, and is able to run these jobs as foreground services if needed. In many cases, using WorkManager is preferable to using foreground services directly.

Background

A background service performs an operation that isn't directly noticed by the user. For example, if an app used a service to compact its storage, that would usually be a background service.

> ★ **Note:** If your app targets API level 26 or higher, the system imposes restrictions on running background services when the app itself isn't in the foreground. In most situations, for example, you shouldn't access location information from the background. Instead, schedule tasks using WorkManager.

Bound

A service is *bound* when an application component binds to it by calling `bindService()`. A bound service offers a client-server interface that allows components to interact with the service, send requests, receive results, and even do so across processes with interprocess communication (IPC). A bound service runs only as long as another application component is bound to it. Multiple components can bind to the service at once, but when all of them unbind, the service is destroyed.

Claim	Public Documentation
	; https://developer.android.com/guide/components/activities/activity-lifecycle:

Activity-lifecycle concepts

To navigate transitions between stages of the activity lifecycle, the `Activity` class provides a core set of six callbacks: `onCreate()`, `onStart()`, `onResume()`, `onPause()`, `onStop()`, and `onDestroy()`. The system invokes each of these callbacks as the activity enters a new state.

Figure 1 presents a visual representation of this paradigm.

As the user begins to leave the activity, the system calls methods to dismantle the activity. In some cases, the activity is only partially dismantled and still resides in memory, such as when the user switches to another app. In these cases, the activity can still come back to the foreground.

If the user returns to the activity, it resumes from where the user left off. With a few exceptions, apps are restricted from starting activities when running in the background.

The system's likelihood of killing a given process, along with the activities in it, depends on the state of the activity at the time. For more information on the relationship between state and vulnerability to ejection, see the section about activity state and ejection from memory.

Depending on the complexity of your activity, you probably don't need to implement all the lifecycle methods. However, it's important that you understand each one and implement those that make your app behave the way users expect.

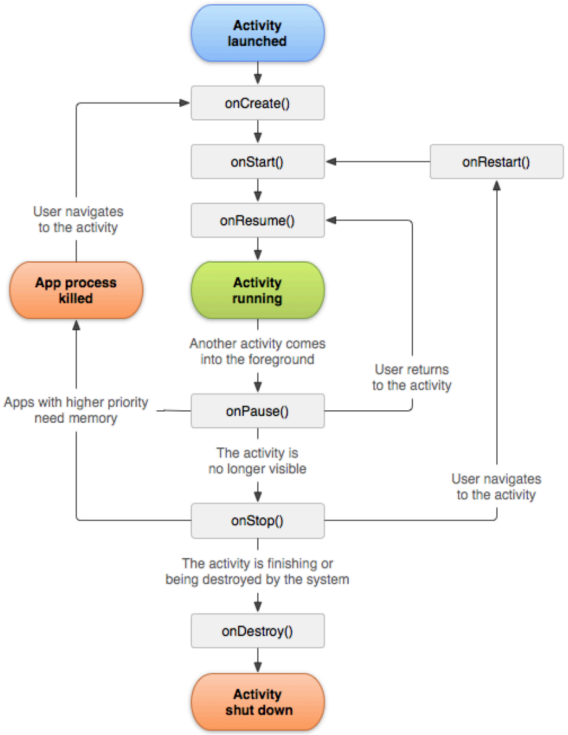


Figure 1. A simplified illustration of the activity lifecycle.

; https://developer.android.com/guide/components/activities/intro-activities.

Claim	Public Documentation
[1j] is interacting with the user in the device user interface foreground, and	The Accused Instrumentalities comprise one or more applications "interacting with the user in the device user interface foreground." For example, Google's devices, including the Google Pixel 7 Pro, sold and used by T-Mobile classify applications and internet service activities in both foreground and background. *See, e.g.*, https://support.google.com/pixelphone/answer/2819524?sjid=13223854186446774975-NC#zippy=: **Reduce and manage mobile data usage** Data usage is how much data your phone uploads or downloads using mobile data. To make sure that you're not using too much data on your data plan, you can check and change your data usage. **Important:** Some of these steps work only on Android 8.0 and up. Learn how to check your Android version. **Check your mobile data usage** 1. Open your phone's Settings app. 2. Tap **Network and Internet** > **Internet**. 3. Next to your operator, tap Settings ⚙. 4. At the top you'll see how much total data you use. 5. To see graphs and details, tap **App data usage**. • To pick a time period, tap the down arrow ⌄. • To see how much data each app uses, look below the graph. https://support.google.com/pixelphone/answer/7055392?hl=en-AU&sjid=13223854186446774975-NC:

Claim	Public Documentation

	# Use less mobile data with Data Saver
	
	To help use less mobile data on a limited data plan, you can turn on Data Saver. This mode lets most apps and services get background data only via Wi-Fi. Currently active apps and services can use mobile data.
	
	Important: Some of these steps work only on Android 8.0 and up. Learn how to check your Android version.
	
	## Turn Data Saver on or off
	
	1. Open your phone's Settings app.
	2. Tap **Network and Internet** › Data Saver.
	3. Turn **Data Saver** on or off.
	   • In your status bar, when Data Saver is on, you'll see the Data Saver icon .
	   • You'll also see a notification at the top of your phone's Settings app.
	
	Tip: You can add Data Saver to your settings bar. Learn how to customise Quick Settings.
	
	; https://support.google.com/pixelphone/answer/6187458?hl=en&sjid=13223854186446774975-NC:

Use Battery Saver on a Pixel phone

You can set Battery Saver to turn on automatically when your phone's battery gets low. You can also turn on Battery Saver at any time. To save even more power on your Pixel 3 or later phone, including Fold, you can turn on Extreme Battery Saver.

Important: While Battery Saver is on, a Pixel phone with 5G uses 4G service. Learn what changes while Battery Saver is on.

Important: Some of these steps work only on Android 11 and up. Learn how to check your Android version.

Turn Battery Saver on or off

When Battery Saver is on, at the top of your screen, you'll find Battery Saver on . To turn Battery Saver on or off:

1. At the top of your screen, swipe down.
2. Tap Battery Saver .

Turn Battery Saver on automatically

1. Open your phone's Settings app.
2. Tap **Battery** > **Battery Saver** > **Schedule and reminders.**
3. Make sure that "Turn on based on battery level" is on.
4. Slide the percentage bar to the level you want.

Tip: The next time your battery level falls to the percentage you have set, Battery Saver will turn on automatically for you.

Claim	Public Documentation
	; https://support.google.com/pixelphone/answer/7015477?hl=en&sjid=13223854186446774975-NC:

Keep Adaptive Battery & battery optimization on

To have apps use your phone's battery only when you need them to, keep Adaptive Battery and battery optimization on. These settings are on by default.

Important: Some of these steps work only on Android 9.0 and up. Learn how to check your Android version.

The Pixel battery continuously learns your behaviors and optimizes itself based on your most recent app usage. When you set up a new device or after a factory reset, optimization may take up a few weeks to take full effect. For best results keep Adaptive Battery and Battery Optimization on .

It's normal for your Pixel battery to drain a little more than usual after a software update. This is because the phone is working hard to download and optimize the new software and get everything up and running.

If you still experience unusual battery drain after a few days, please let us know and we'll be happy to help.

Check that Adaptive Battery is on for your phone

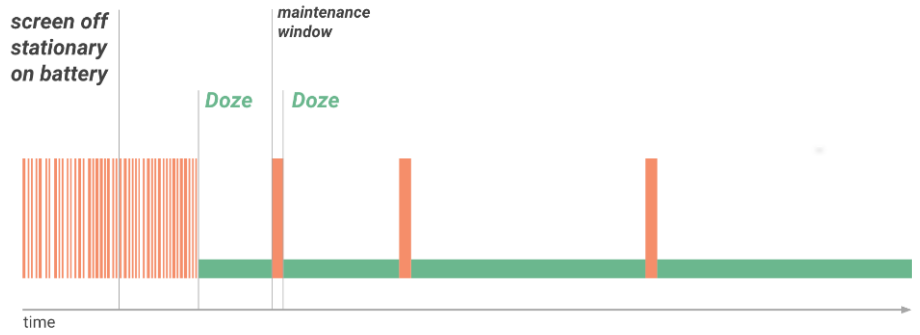
If you keep Adaptive Battery on, apps that you use less often will run less while you're not using them. Your phone can learn how you use your apps over time. This can help save battery in ways that work best for you.

1. Open your phone's Settings app.
2. Tap **Battery** > **Battery Saver**.
3. Tap **Adaptive Battery** > Expand More .
4. Turn on **Use Adaptive Battery** if it's turned off.

Tip: Adaptive Battery learns from your phone usage to continuously optimize how apps use battery. To extend battery life, it may reduce performance and delay notifications.

Claim	Public Documentation

	; https://developer.android.com/training/basics/network-ops/data-saver:
	
	Optimize network data usage
	
	Over the life of a smartphone, the cost of a cellular data plan can easily exceed the cost of the device itself. On Android 7.0 (API level 24) and higher, users can enable Data Saver on a device-wide basis in order to optimize their device's data usage, and use less data. This ability is especially useful when roaming, near the end of the billing cycle, or for a small prepaid data pack.
	
	When a user enables Data Saver in **Settings** and the device is on a metered network, the system blocks background data usage and signals apps to use less data in the foreground wherever possible. Users can allow specific apps to use background metered data usage even when Data Saver is turned on.
	
	Android 7.0 (API level 24) extends the `ConnectivityManager` API to provide apps with a way to retrieve the user's Data Saver preferences and monitor preference changes. It is considered good practice for apps to check whether the user has enabled Data Saver and make an effort to limit foreground and background data usage.
	
	Check data saver preferences
	
	On Android 7.0 (API level 24) and higher, apps can use the `ConnectivityManager` API to determine what data usage restrictions are being applied. The `getRestrictBackgroundStatus()` method returns one of the following values:
	
	`RESTRICT_BACKGROUND_STATUS_DISABLED`
	Data Saver is disabled.
	
	`RESTRICT_BACKGROUND_STATUS_ENABLED`
	The user has enabled Data Saver for this app. Apps should make an effort to limit data usage in the foreground and gracefully handle restrictions to background data usage.
	
	`RESTRICT_BACKGROUND_STATUS_WHITELISTED`
	The user has enabled Data Saver but the app is allowed to bypass it. Apps should still make an effort to limit foreground and background data usage.
	
	Limit data usage whenever the device is connected to a metered network, even if Data Saver is disabled or the app is allowed to bypass it. The following sample code uses `ConnectivityManager.isActiveNetworkMetered()` and `ConnectivityManager.getRestrictBackgroundStatus()` to determine how much data the app should use:
	
	; https://developer.android.com/training/monitoring-device-state/doze-standby:

Claim	Public Documentation
	Optimize for Doze and App Standby Starting from Android 6.0 (API level 23), Android introduces two power-saving features that extend battery life for users by managing how apps behave when a device is not connected to a power source. *Doze* reduces battery consumption by deferring background CPU and network activity for apps when the device is unused for long periods of time. *App Standby* defers background network activity for apps with which the user has not recently interacted. While the device is in Doze, apps' access to certain battery-intensive resources is deferred until maintenance windows. The specific restrictions are listed in Power Management Restrictions. Doze and App Standby manage the behavior of all apps running on Android 6.0 or higher, regardless whether they are specifically targeting API level 23. To ensure the best experience for users, test your app in Doze and App Standby modes and make any necessary adjustments to your code. The sections below provide details. **Understanding Doze** If a user leaves a device unplugged and stationary for a period of time, with the screen off, the device enters Doze mode. In Doze mode, the system attempts to conserve battery by restricting apps' access to network and CPU-intensive services. It also prevents apps from accessing the network and defers their jobs, syncs, and standard alarms. Periodically, the system exits Doze for a brief time to let apps complete their deferred activities. During this *maintenance window*, the system runs all pending syncs, jobs, and alarms, and lets apps access the network.  **Figure 1.** Doze provides a recurring maintenance window for apps to use the network and handle pending activities.

Claim	Public Documentation
	At the conclusion of each maintenance window, the system again enters Doze, suspending network access and deferring jobs, syncs, and alarms. Over time, the system schedules maintenance windows less and less frequently, helping to reduce battery consumption in cases of longer-term inactivity when the device is not connected to a charger. As soon as the user wakes the device by moving it, turning on the screen, or connecting a charger, the system exits Doze and all apps return to normal activity. The Doze restriction on network access is also likely to affect your app, especially if the app relies on real-time messages such as tickles or notifications. If your app requires a persistent connection to the network to receive messages, you should use Firebase Cloud Messaging (FCM) if possible. ; https://developer.android.com/topic/performance/appstandby:

App Standby Buckets

Android 9 (API level 28) and higher support **App Standby Buckets**. App Standby Buckets help the system prioritize apps' requests for resources based on how recently and how frequently the apps are used. Based on app usage patterns, each app is placed in one of five priority **buckets**. The system limits the device resources available to each app based on which bucket the app is in.

Priority buckets

The system dynamically assigns each app to a priority bucket, reassigning the apps as needed. The system may rely on a preloaded app that uses machine learning to determine how likely each app is to be used, and assigns apps to the appropriate buckets. If the system app is not present on a device, the system defaults to sorting apps based on how recently they were used. More active apps are assigned to buckets that give the apps higher priority, making more system resources available to the app. In particular, the bucket determines how frequently the app's jobs run, and how often the app can trigger alarms. These restrictions apply only while the device is on battery power; the system does not impose these restrictions on apps while the device is charging.

> **Note:** Every manufacturer can set their own criteria for how non-active apps are assigned to buckets. You should not try to influence which bucket your app is assigned to. Instead, focus on making sure your app behaves well in whatever bucket it might be in. Your app can find out what bucket it's currently in by calling `UsageStatsManager.getAppStandbyBucket()`.

The buckets are:

1. Active: App is currently being used or was very recently used.
2. Working set: App is in regular use.
3. Frequent: App is often used, but not every day.
4. Rare: App is not frequently used.
5. Restricted: App consumes a great deal of system resources, or may exhibit undesirable behavior.

In addition, there's a special **never** bucket for apps that have been installed but have never been run. The system imposes severe restrictions on these apps.

Claim	Public Documentation
	; https://developer.android.com/topic/performance/power/power-details; https://developer.android.com/topic/performance/background-optimization; https://developer.android.com/reference/android/app/job/JobScheduler; https://developer.android.com/guide/background/persistent; https://developer.android.com/guide/components/activities/process-lifecycle: 1. A **foreground process** is one that is required for what the user is currently doing. Various application components can cause its containing process to be considered foreground in different ways. A process is considered to be in the foreground if any of the following conditions hold: • It is running an `Activity` at the top of the screen that the user is interacting with (its `onResume()` method has been called). • It has a `BroadcastReceiver` that is currently running (its `BroadcastReceiver.onReceive()` method is executing). • It has a `Service` that is currently executing code in one of its callbacks (`Service.onCreate()`, `Service.onStart()`, or `Service.onDestroy()`). There will only ever be a few such processes in the system, and these will only be killed as a last resort if memory is so low that not even these processes can continue to run. Generally, at this point, the device has reached a memory paging state, so this action is required in order to keep the user interface responsive. ; https://developer.android.com/guide/background:

Claim	Public Documentation
	Definition of background work An app is running in the *background* when both the following conditions are satisfied: • None of the app's activities are currently visible to the user. • The app isn't running any foreground services that started while an activity from the app was visible to the user. Otherwise, the app is running in the *foreground*. ; https://developer.android.com/guide/components/services;

Types of Services

These are the three different types of services:

Foreground

A foreground service performs some operation that is noticeable to the user. For example, an audio app would use a foreground service to play an audio track. Foreground services must display a Notification. Foreground services continue running even when the user isn't interacting with the app.

When you use a foreground service, you must display a notification so that users are actively aware that the service is running. This notification cannot be dismissed unless the service is either stopped or removed from the foreground.

Learn more about how to configure foreground services in your app.

> **Note:** The WorkManager API offers a flexible way of scheduling tasks, and is able to run these jobs as foreground services if needed. In many cases, using WorkManager is preferable to using foreground services directly.

Background

A background service performs an operation that isn't directly noticed by the user. For example, if an app used a service to compact its storage, that would usually be a background service.

> **Note:** If your app targets API level 26 or higher, the system imposes restrictions on running background services when the app itself isn't in the foreground. In most situations, for example, you shouldn't access location information from the background. Instead, schedule tasks using WorkManager.

Bound

A service is *bound* when an application component binds to it by calling `bindService()`. A bound service offers a client-server interface that allows components to interact with the service, send requests, receive results, and even do so across processes with interprocess communication (IPC). A bound service runs only as long as another application component is bound to it. Multiple components can bind to the service at once, but when all of them unbind, the service is destroyed.

Claim	Public Documentation
	; https://developer.android.com/guide/components/activities/intro-activities.
[1k] selectively allow or deny one or more Internet service activities by or on behalf of the particular application based on whether or not the particular application is one of the first one or more applications, the differential traffic control policy, including any applicable user augmentation of the differential traffic control policy, and the classifications performed by the one or more processors.	The Accused Instrumentalities "selectively allow or deny one or more Internet service activities by or on behalf of the particular application based on whether or not the particular application is one of the first one or more applications, the differential traffic control policy, including any applicable user augmentation of the differential traffic control policy, and the classifications performed by the one or more processors." For example, Google's devices, including the Google Pixel 7 Pro, sold and used by T-Mobile allow or deny internet service activities by or on behalf of applications based on classifications of particular applications and policies. *See, e.g.*, https://support.google.com/pixelphone/answer/2819524?sjid=13223854186446774975-NC#zippy=:

Claim	Public Documentation

	# Reduce and manage mobile data usage
	
	Data usage is how much data your phone uploads or downloads using mobile data. To make sure that you're not using too much data on your data plan, you can check and change your data usage.
	
	Important: Some of these steps work only on Android 8.0 and up. Learn how to check your Android version.
	
	## Check your mobile data usage
	
	1. Open your phone's Settings app.
	2. Tap **Network and Internet** > **Internet**.
	3. Next to your operator, tap Settings ⚙.
	4. At the top you'll see how much total data you use.
	5. To see graphs and details, tap **App data usage**.
	   • To pick a time period, tap the down arrow ⌄.
	   • To see how much data each app uses, look below the graph.
	
	https://support.google.com/pixelphone/answer/7055392?hl=en-AU&sjid=13223854186446774975-NC:

Claim	Public Documentation

	# Use less mobile data with Data Saver
	
	To help use less mobile data on a limited data plan, you can turn on Data Saver. This mode lets most apps and services get background data only via Wi-Fi. Currently active apps and services can use mobile data.
	
	Important: Some of these steps work only on Android 8.0 and up. Learn how to check your Android version.
	
	## Turn Data Saver on or off
	
	1. Open your phone's Settings app.
	2. Tap **Network and Internet** › Data Saver.
	3. Turn **Data Saver** on or off.
	   • In your status bar, when Data Saver is on, you'll see the Data Saver icon .
	   • You'll also see a notification at the top of your phone's Settings app.
	
	Tip: You can add Data Saver to your settings bar. Learn how to customise Quick Settings.
	
	; https://support.google.com/pixelphone/answer/6187458?hl=en&sjid=13223854186446774975-NC:

Use Battery Saver on a Pixel phone

You can set Battery Saver to turn on automatically when your phone's battery gets low. You can also turn on Battery Saver at any time. To save even more power on your Pixel 3 or later phone, including Fold, you can turn on Extreme Battery Saver.

Important: While Battery Saver is on, a Pixel phone with 5G uses 4G service. Learn what changes while Battery Saver is on.

Important: Some of these steps work only on Android 11 and up. Learn how to check your Android version.

Turn Battery Saver on or off

When Battery Saver is on, at the top of your screen, you'll find Battery Saver on . To turn Battery Saver on or off:

1. At the top of your screen, swipe down.
2. Tap Battery Saver .

Turn Battery Saver on automatically

1. Open your phone's Settings app.
2. Tap **Battery** > **Battery Saver** > **Schedule and reminders.**
3. Make sure that "Turn on based on battery level" is on.
4. Slide the percentage bar to the level you want.

Tip: The next time your battery level falls to the percentage you have set, Battery Saver will turn on automatically for you.

Claim	Public Documentation
	; https://support.google.com/pixelphone/answer/7015477?hl=en&sjid=13223854186446774975-NC:

Keep Adaptive Battery & battery optimization on

To have apps use your phone's battery only when you need them to, keep Adaptive Battery and battery optimization on. These settings are on by default.

Important: Some of these steps work only on Android 9.0 and up. Learn how to check your Android version.

The Pixel battery continuously learns your behaviors and optimizes itself based on your most recent app usage. When you set up a new device or after a factory reset, optimization may take up a few weeks to take full effect. For best results keep Adaptive Battery and Battery Optimization on .

It's normal for your Pixel battery to drain a little more than usual after a software update. This is because the phone is working hard to download and optimize the new software and get everything up and running.

If you still experience unusual battery drain after a few days, please let us know and we'll be happy to help.

Check that Adaptive Battery is on for your phone

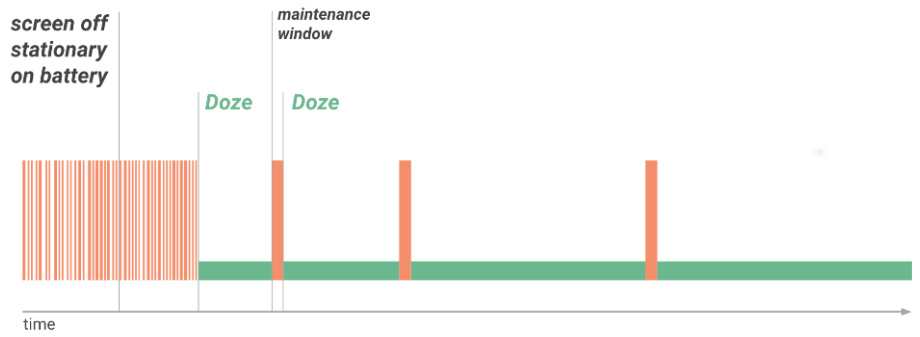
If you keep Adaptive Battery on, apps that you use less often will run less while you're not using them. Your phone can learn how you use your apps over time. This can help save battery in ways that work best for you.

1. Open your phone's Settings app.
2. Tap **Battery** > **Battery Saver**.
3. Tap **Adaptive Battery** > Expand More .
4. Turn on **Use Adaptive Battery** if it's turned off.

Tip: Adaptive Battery learns from your phone usage to continuously optimize how apps use battery. To extend battery life, it may reduce performance and delay notifications.

Claim	Public Documentation

	; https://developer.android.com/training/basics/network-ops/data-saver:
	
	Optimize network data usage
	
	Over the life of a smartphone, the cost of a cellular data plan can easily exceed the cost of the device itself. On Android 7.0 (API level 24) and higher, users can enable Data Saver on a device-wide basis in order to optimize their device's data usage, and use less data. This ability is especially useful when roaming, near the end of the billing cycle, or for a small prepaid data pack.
	
	When a user enables Data Saver in **Settings** and the device is on a metered network, the system blocks background data usage and signals apps to use less data in the foreground wherever possible. Users can allow specific apps to use background metered data usage even when Data Saver is turned on.
	
	Android 7.0 (API level 24) extends the `ConnectivityManager` API to provide apps with a way to retrieve the user's Data Saver preferences and monitor preference changes. It is considered good practice for apps to check whether the user has enabled Data Saver and make an effort to limit foreground and background data usage.
	
	Check data saver preferences
	
	On Android 7.0 (API level 24) and higher, apps can use the `ConnectivityManager` API to determine what data usage restrictions are being applied. The `getRestrictBackgroundStatus()` method returns one of the following values:
	
	`RESTRICT_BACKGROUND_STATUS_DISABLED`
	
	Data Saver is disabled.
	
	`RESTRICT_BACKGROUND_STATUS_ENABLED`
	
	The user has enabled Data Saver for this app. Apps should make an effort to limit data usage in the foreground and gracefully handle restrictions to background data usage.
	
	`RESTRICT_BACKGROUND_STATUS_WHITELISTED`
	
	The user has enabled Data Saver but the app is allowed to bypass it. Apps should still make an effort to limit foreground and background data usage.
	
	Limit data usage whenever the device is connected to a metered network, even if Data Saver is disabled or the app is allowed to bypass it. The following sample code uses `ConnectivityManager.isActiveNetworkMetered()` and `ConnectivityManager.getRestrictBackgroundStatus()` to determine how much data the app should use:
	
	; https://developer.android.com/training/monitoring-device-state/doze-standby:

Claim	Public Documentation
	Optimize for Doze and App Standby Starting from Android 6.0 (API level 23), Android introduces two power-saving features that extend battery life for users by managing how apps behave when a device is not connected to a power source. *Doze* reduces battery consumption by deferring background CPU and network activity for apps when the device is unused for long periods of time. *App Standby* defers background network activity for apps with which the user has not recently interacted. While the device is in Doze, apps' access to certain battery-intensive resources is deferred until maintenance windows. The specific restrictions are listed in Power Management Restrictions. Doze and App Standby manage the behavior of all apps running on Android 6.0 or higher, regardless whether they are specifically targeting API level 23. To ensure the best experience for users, test your app in Doze and App Standby modes and make any necessary adjustments to your code. The sections below provide details. **Understanding Doze** If a user leaves a device unplugged and stationary for a period of time, with the screen off, the device enters Doze mode. In Doze mode, the system attempts to conserve battery by restricting apps' access to network and CPU-intensive services. It also prevents apps from accessing the network and defers their jobs, syncs, and standard alarms. Periodically, the system exits Doze for a brief time to let apps complete their deferred activities. During this *maintenance window*, the system runs all pending syncs, jobs, and alarms, and lets apps access the network.  **Figure 1.** Doze provides a recurring maintenance window for apps to use the network and handle pending activities.

Claim	Public Documentation
	At the conclusion of each maintenance window, the system again enters Doze, suspending network access and deferring jobs, syncs, and alarms. Over time, the system schedules maintenance windows less and less frequently, helping to reduce battery consumption in cases of longer-term inactivity when the device is not connected to a charger. As soon as the user wakes the device by moving it, turning on the screen, or connecting a charger, the system exits Doze and all apps return to normal activity. The Doze restriction on network access is also likely to affect your app, especially if the app relies on real-time messages such as tickles or notifications. If your app requires a persistent connection to the network to receive messages, you should use Firebase Cloud Messaging (FCM) if possible. ; https://developer.android.com/topic/performance/appstandby:

App Standby Buckets

Android 9 (API level 28) and higher support **App Standby Buckets**. App Standby Buckets help the system prioritize apps' requests for resources based on how recently and how frequently the apps are used. Based on app usage patterns, each app is placed in one of five priority **buckets**. The system limits the device resources available to each app based on which bucket the app is in.

Priority buckets

The system dynamically assigns each app to a priority bucket, reassigning the apps as needed. The system may rely on a preloaded app that uses machine learning to determine how likely each app is to be used, and assigns apps to the appropriate buckets. If the system app is not present on a device, the system defaults to sorting apps based on how recently they were used. More active apps are assigned to buckets that give the apps higher priority, making more system resources available to the app. In particular, the bucket determines how frequently the app's jobs run, and how often the app can trigger alarms. These restrictions apply only while the device is on battery power; the system does not impose these restrictions on apps while the device is charging.

> **Note:** Every manufacturer can set their own criteria for how non-active apps are assigned to buckets. You should not try to influence which bucket your app is assigned to. Instead, focus on making sure your app behaves well in whatever bucket it might be in. Your app can find out what bucket it's currently in by calling `UsageStatsManager.getAppStandbyBucket()`.

The buckets are:

1. Active: App is currently being used or was very recently used.
2. Working set: App is in regular use.
3. Frequent: App is often used, but not every day.
4. Rare: App is not frequently used.
5. Restricted: App consumes a great deal of system resources, or may exhibit undesirable behavior.

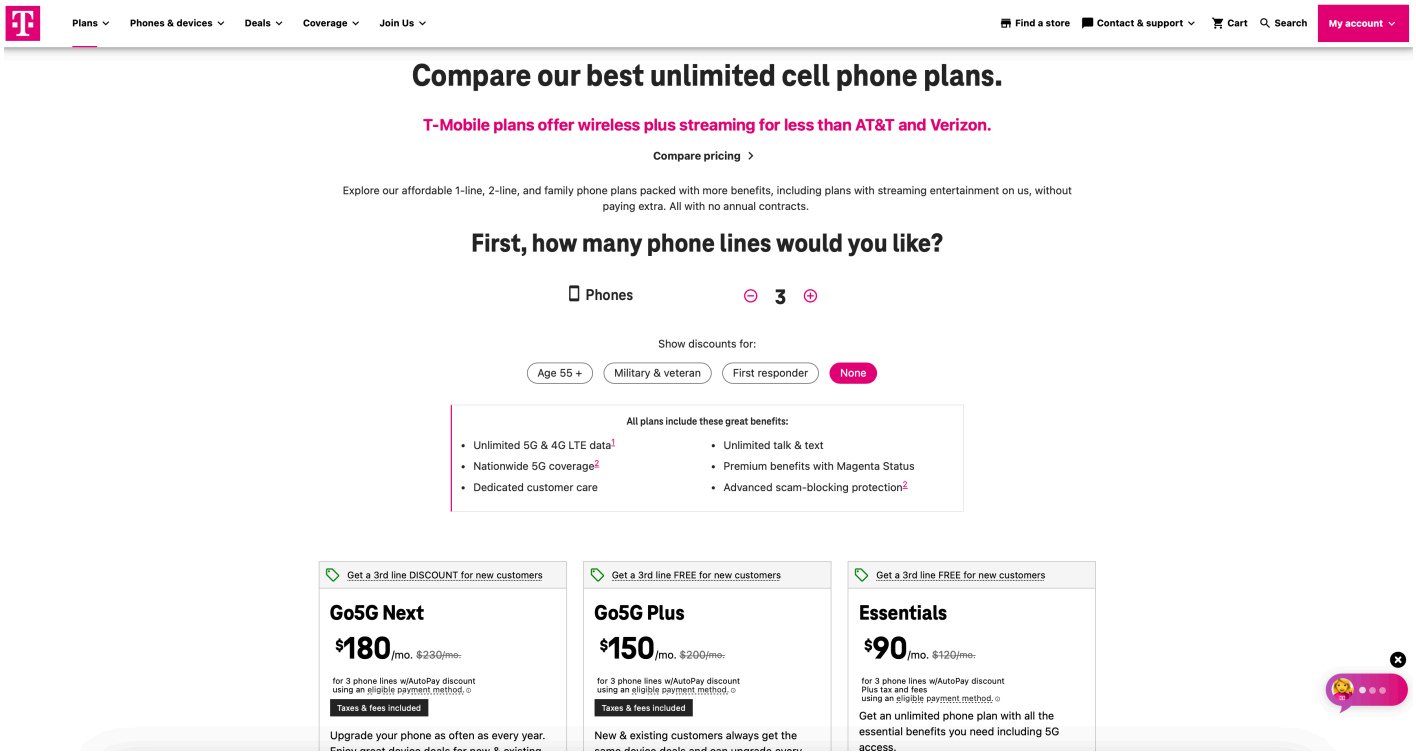
In addition, there's a special **never** bucket for apps that have been installed but have never been run. The system imposes severe restrictions on these apps.

Claim	Public Documentation
	; https://developer.android.com/topic/performance/power/power-details; https://developer.android.com/topic/performance/background-optimization; https://developer.android.com/reference/android/app/job/JobScheduler; https://developer.android.com/guide/background/persistent; https://developer.android.com/guide/components/activities/process-lifecycle; https://developer.android.com/guide/background; https://developer.android.com/about/versions/pie/android-9.0; https://developer.android.com/training/basics/network-ops/reading-network-state; https://developer.android.com/training/connectivity/network-access-optimization; https://developer.android.com/reference/android/net/NetworkCapabilities.
2. The wireless end-user device of claim 1, wherein based on the differential traffic control policy the one or more processors selectively deny one or more Internet service activities by or on behalf of the particular application when the particular application is one of the first one or more applications, the classified wireless network is a WWAN type, and the particular application is classified as not interacting with the user in the device user interface foreground.	The Accused Instrumentalities comprise "[t]he wireless end-user device of claim 1, wherein based on the differential traffic control policy the one or more processors selectively deny one or more Internet service activities by or on behalf of the particular application when the particular application is one of the first one or more applications, the classified wireless network is a WWAN type, and the particular application is classified as not interacting with the user in the device user interface foreground." *See*, for example, the disclosures identified for claim 1.
3. The wireless end-user device of claim 2, wherein the one or more processors are further configured to override the selective denial of one or more Internet service activities by or on behalf of the particular application when the user has augmented the differential traffic control policy so as to indicate that	The Accused Instrumentalities comprise "[t]he wireless end-user device of claim 2, wherein the one or more processors are further configured to override the selective denial of one or more Internet service activities by or on behalf of the particular application when the user has augmented the differential traffic control policy so as to indicate that Internet service activities are allowable when the classified wireless network is the WWAN type, and the particular application is classified as not interacting with the user in the device user interface foreground." *See*, for example, the disclosures identified for claims 1-2.

Claim	Public Documentation
Internet service activities are allowable when the classified wireless network is the WWAN type, and the particular application is classified as not interacting with the user in the device user interface foreground.	
4. The wireless end-user device of claim 2, wherein based on the differential traffic control policy the one or more processors selectively allow one or more Internet service activities by or on behalf of the particular application when the particular application is one of the first one or more applications, the classified wireless network is the WWAN type, and the particular application is classified as interacting with the user in the device user interface foreground.	The Accused Instrumentalities comprise "[t]he wireless end-user device of claim 2, wherein based on the differential traffic control policy the one or more processors selectively allow one or more Internet service activities by or on behalf of the particular application when the particular application is one of the first one or more applications, the classified wireless network is the WWAN type, and the particular application is classified as interacting with the user in the device user interface foreground." *See*, for example, the disclosures identified for claims 1-2.
5. The wireless end-user device of claim 1, wherein based on the differential traffic control policy the one or more processors selectively allow one or more Internet service activities by or on behalf of a second particular application and/or service when the second particular application and/or service is one of the second one or	The Accused Instrumentalities comprise "[t]he wireless end-user device of claim 1, wherein based on the differential traffic control policy the one or more processors selectively allow one or more Internet service activities by or on behalf of a second particular application and/or service when the second particular application and/or service is one of the second one or more applications and/or services and the classified wireless network is the WWAN type." *See*, for example, the disclosures identified for claim 1.

Claim	Public Documentation
more applications and/or services and the classified wireless network is the WWAN type.	
6. The wireless end-user device of claim 1, wherein the one or more processors are configured to classify that the particular application is interacting with the user in the device user interface foreground when the user of the device is directly interacting with that application or perceiving any benefit from that application.	The Accused Instrumentalities comprise "[t]he wireless end-user device of claim 1, wherein the one or more processors are configured to classify that the particular application is interacting with the user in the device user interface foreground when the user of the device is directly interacting with that application or perceiving any benefit from that application." *See*, for example, the disclosures identified for claim 1.
7. The wireless end-user device of claim 1, wherein the user interface is further to provide the user of the device with information regarding why the differential traffic control policy is applied to the particular application.	The Accused Instrumentalities comprise "[t]he wireless end-user device of claim 1, wherein the user interface is further to provide the user of the device with information regarding why the differential traffic control policy is applied to the particular application." *See*, for example, the disclosures identified for claim 1.
8. The wireless end-user device of claim 1, wherein the differential traffic control policy is part of a multimode profile having different policies for different ones of the network types.	The Accused Instrumentalities comprise "[t]he wireless end-user device of claim 1, wherein the differential traffic control policy is part of a multimode profile having different policies for different ones of the network types." *See*, for example, the disclosures identified for claim 1.
9. The wireless end-user device of claim 8, wherein the one or more processors are further configured	The Accused Instrumentalities comprise "[t]he wireless end-user device of claim 8, wherein the one or more processors are further configured to select a traffic control policy from the multimode profile based at least in part on the classified wireless network type."

Claim	Public Documentation
to select a traffic control policy from the multimode profile based at least in part on the classified wireless network type.	*See*, for example, the disclosures identified for claims 1 and 8.
10. The wireless end-user device of claim 9, wherein the one or more processors are further configured to, when the classified wireless network type is at least one type of WLAN, select the traffic control policy from the multimode profile based at least in part on a type of network connection from the WLAN to the Internet.	The Accused Instrumentalities comprise "[t]he wireless end-user device of claim 9, wherein the one or more processors are further configured to, when the classified wireless network type is at least one type of WLAN, select the traffic control policy from the multimode profile based at least in part on a type of network connection from the WLAN to the Internet." *See*, for example, the disclosures identified for claim 1 and 9.
11. The wireless end-user device of claim 1, wherein the plurality of network types include three or more of 2G, 3G, 4G, home, roaming, and WiFi.	The Accused Instrumentalities comprise "[t]he wireless end-user device of claim 1, wherein the plurality of network types include three or more of 2G, 3G, 4G, home, roaming, and WiFi." *See*, for example, the disclosures identified for claim 1.
12. The wireless end-user device of claim 1, the one or more processors further configured to receive an update to at least a portion of the differential traffic control policy list from a network element.	The Accused Instrumentalities comprise "[t]he wireless end-user device of claim 1, the one or more processors further configured to receive an update to at least a portion of the differential traffic control policy list from a network element." *See*, for example, the disclosures identified for claim 1. As yet another example, the one or more processors are configured to receive portions of policies from a network element. *See, e.g.,* https://www.t-mobile.com/cell-phone-plans:

Claim	Public Documentation
	 ; https://www.t-mobile.com/cell-phone-plans/affordable-data-plans; https://www.t-mobile.com/cell-phone-plans/unlimited-55-senior-discount-plans?INTNAV=tNav:Plans:UnlimitedAge55; https://www.t-mobile.com/cell-phone-plans/military-discount-plans; https://www.t-mobile.com/cell-phone-plans/first-responder-discounts; https://www.t-mobile.com/home-internet/plans; ; https://developer.android.com/about/versions/pie/android-9.0:

Claim	Public Documentation
	Data cost sensitivity in JobScheduler Beginning in Android 9, `JobScheduler` can use network status signals provided by carriers to improve the handling of network-related jobs. Jobs can declare their estimated data size, signal prefetching, and specify detailed network requirements. `JobScheduler` then manages work according to the network status. For example, when the network signals that it is congested, `JobScheduler` might defer large network requests. When on an unmetered network, `JobScheduler` can run prefetch jobs to improve the user experience, such as by prefetching headlines. When adding jobs, make sure to use `setEstimatedNetworkBytes()`, `setPrefetch()`, and `setRequiredNetwork()` when appropriate to help `JobScheduler` handle the work properly. When your job executes, be sure to use the `Network` object returned by `JobParameters.getNetwork()`. Otherwise you'll implicitly use the device's default network which may not meet your requirements, causing unintended data usage. ; https://developer.android.com/training/basics/network-ops/reading-network-state; https://developer.android.com/training/connectivity/network-access-optimization; https://developer.android.com/reference/android/net/NetworkCapabilities.
13. The wireless end-user device of claim 1, wherein the plurality of network types include a roaming WWAN type and a home WWAN type, and the one or more processors are to apply the differential traffic control policy to one of but not both of the roaming WWAN type and the home WWAN type.	The Accused Instrumentalities comprise "[t]he wireless end-user device of claim 1, wherein the plurality of network types include a roaming WWAN type and a home WWAN type, and the one or more processors are to apply the differential traffic control policy to one of but not both of the roaming WWAN type and the home WWAN type." *See*, for example, the disclosures identified for claim 1. For further example, the policy can be based on roaming on a WWAN network. *See, e.g.,* https://www.t-mobile.com/support/coverage/domestic-roaming-data:

SUPPORT › COVERAGE

Domestic roaming data

Data works a little differently when connected outside the T-Mobile network in the U.S. T-Mobile continues to invest billions in expanding network coverage and improving its network speed and performance. In locations in the U.S. where we do not yet have network coverage, we partner with other networks.

On this page:

- How it works
- How much domestic roaming data do you get?
- Check and reduce data use
- What happens when your domestic roaming data is used
- FAQs

How it works

When you travel outside of T-Mobile's U.S. network areas, your phone automatically switches to use one of our wireless network partners where available when you have data roaming enabled.

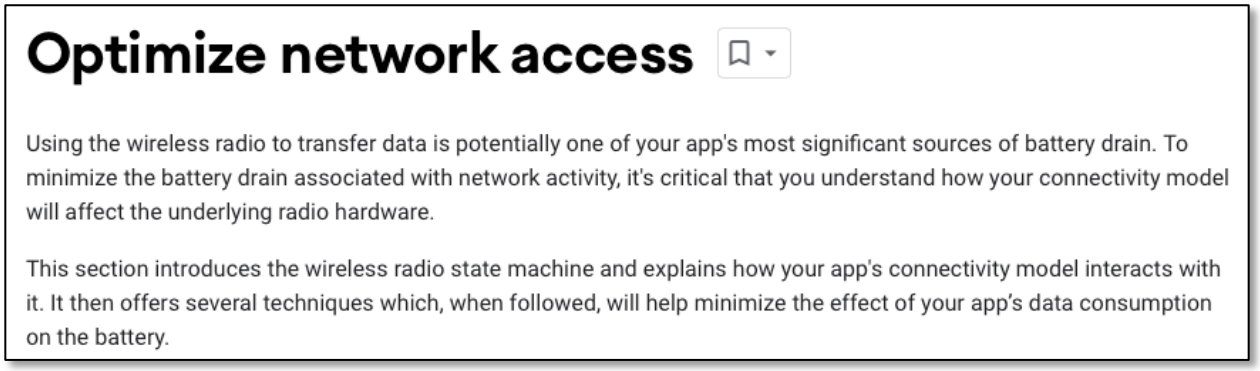
- Check out our map of the network and roaming areas.
- T-Mobile coordinates with these partners to give our customers connectivity outside of our network. T-Mobile does not charge an additional fee for this service, but because we do not own these networks, there are limitations to data use.
- There may be times when your device still attempts to roam on another U.S. wireless network, even when you're within the T-Mobile coverage area. If you'd like to limit this, try the tips to reduce data usage.

How to know if you're roaming domestically

The best way to check your active network is to go into the phone settings and check for the mobile network or phone status options. The process varies by device, and you can find it in your user guide.

- When roaming on these networks, you'll receive free usage alerts via text message to alert you if you approach/reach your available domestic roaming data.
- You can review the T-Mobile coverage map prior to traveling to determine if your destination is within a T-Mobile or partner network area.

Claim	Public Documentation
	; https://www.t-mobile.com/support/coverage/international-roaming-services.
14. The wireless end-user device of claim 1, wherein the plurality of network types include the WWAN type and a WLAN type, and the one or more processors are to apply the differential traffic control policy to one of but not both of the WWAN type and the WLAN type.	The Accused Instrumentalities comprise "[t]he wireless end-user device of claim 1, wherein the plurality of network types include the WWAN type and a WLAN type, and the one or more processors are to apply the differential traffic control policy to one of but not both of the WWAN type and the WLAN type." *See*, for example, the disclosures identified for claim 1.
15. The wireless end-user device of claim 1, wherein the one or more processors are further configured to dynamically change the application of the differential traffic control policy based on a power state of the device.	The Accused Instrumentalities comprise "[t]he wireless end-user device of claim 1, wherein the one or more processors are further configured to dynamically change the application of the differential traffic control policy based on a power state of the device." *See*, for example, the disclosures identified for claim 1.
16. The wireless end-user device of claim 1, wherein the one or more processors are further configured to dynamically change the application of the differential traffic control policy based on a device usage state.	The Accused Instrumentalities comprise "[t]he wireless end-user device of claim 1, wherein the one or more processors are further configured to dynamically change the application of the differential traffic control policy based on a device usage state." *See*, for example, the disclosures identified for claim 1.
17. The wireless end-user device of claim 1, wherein the one or more processors are further configured to dynamically change the applica-	The Accused Instrumentalities comprise "[t]he wireless end-user device of claim 1, wherein the one or more processors are further configured to dynamically change the application of the differential traffic control policy based on power control state changes for one or more of the modems."

Claim	Public Documentation
tion of the differential traffic control policy based on power control state changes for one or more of the modems.	*See*, for example, the disclosures identified for claim 1. As a further example, the one or more processors change policies based on power control state changes of modems. *See, e.g.,* https://developer.android.com/training/connectivity/network-access-optimization.  **Optimize network access** Using the wireless radio to transfer data is potentially one of your app's most significant sources of battery drain. To minimize the battery drain associated with network activity, it's critical that you understand how your connectivity model will affect the underlying radio hardware. This section introduces the wireless radio state machine and explains how your app's connectivity model interacts with it. It then offers several techniques which, when followed, will help minimize the effect of your app's data consumption on the battery.

The radio state machine

The wireless radio on your user's device has built-in power-saving features that help minimize the amount of battery power it consumes. When fully active, the wireless radio consumes significant power, but when inactive or in standby, the radio consumes very little power.

One important factor to remember is that the radio cannot move from standby to fully active instantaneously. There is a latency period associated with "powering up" the radio. So the battery transitions from higher energy states to lower energy states slowly in order to conserve power when not in use while attempting to minimize the latency associated with "powering up" the radio.

The state machine for a typical 3G network radio consists of three energy states:

- **Full power**: Used when a connection is active, allowing the device to transfer data at its highest possible rate.
- **Low power**: An intermediate state that cuts battery power consumption by around 50%.
- **Standby**: The minimal power-consuming state during which no network connection is active.

While the low and standby states drain significantly less battery, they also introduce significant latency to network requests. Returning to full power from the low state takes around 1.5 seconds, and moving from standby to full power can take over 2 seconds.

To minimize latency, the state machine uses a delay to postpone the transition to lower energy states. Figure 1 uses AT&T's timings for a typical 3G radio.

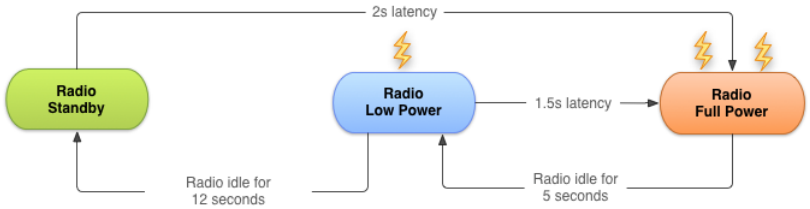


Figure 1. Typical 3G wireless radio state machine.

The radio state machine on each device, particularly the associated transition delay ("tail time") and startup latency, will vary based on the wireless radio technology employed (3G, LTE, 5G, and so on) and is defined and configured by the carrier network over which the device is operating.

This page describes a representative state machine for a typical 3G wireless radio, based on data provided by AT&T. However, the general principles and resulting best practices are applicable for all wireless radio implementations.

This approach is particularly effective for typical mobile web browsing as it prevents unwelcome latency while users browse the web. The relatively low tail-time also ensures that once a browsing session has finished, the radio can move to a lower energy state.

Unfortunately, this approach can lead to inefficient apps on modern smartphone operating systems like Android, where apps run both in the foreground (where latency is important) and in the background (where battery life should be prioritized).

How apps impact the radio state machine

Every time you create a new network connection, the radio transitions to the full power state. In the case of the typical 3G radio state machine described earlier, it will remain at full power for the duration of your transfer—plus an additional 5 seconds of tail time—followed by 12 seconds at the low energy state. So for a typical 3G device, every data transfer session will cause the radio to draw energy for at least 18 seconds.

In practice, this means that an app which makes a one second data transfer, three times a minute, will keep the wireless radio perpetually active, moving it back to high power just as it is entering standby mode.

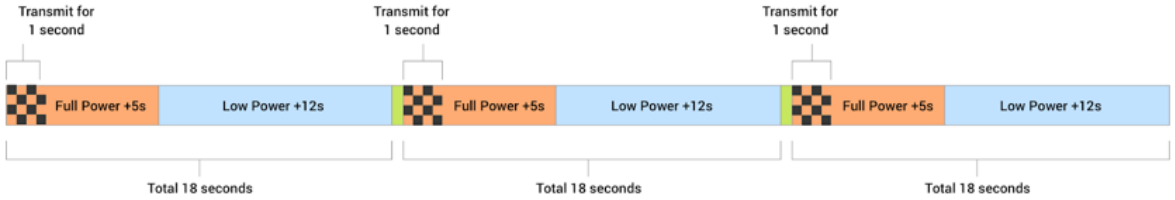


Figure 2. Relative wireless radio power use for one-second transfer running three times every minute. Figure excludes "power up" latency between runs.

By comparison, if the same app bundled its data transfers, running a single three-second transfer every minute, this would keep the radio in the high-power state for a total of only 20 seconds each minute. This would allow the radio to be on standby for 40 seconds of every minute, resulting in a significant reduction in battery consumption.

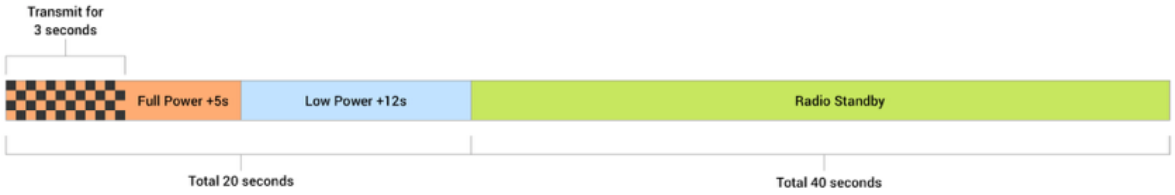


Figure 3. Relative wireless radio power use for three second transfers running once every minute.

Optimization techniques

Now that you understand how network access affects battery life, let's talk about a few things that you can do to help reduce battery drain, while also providing a fast and fluid user experience.

Bundle data transfers

As stated in the previous section, bundling your data transfers so that you're transferring more data less often is one of the best ways to improve battery efficiency.

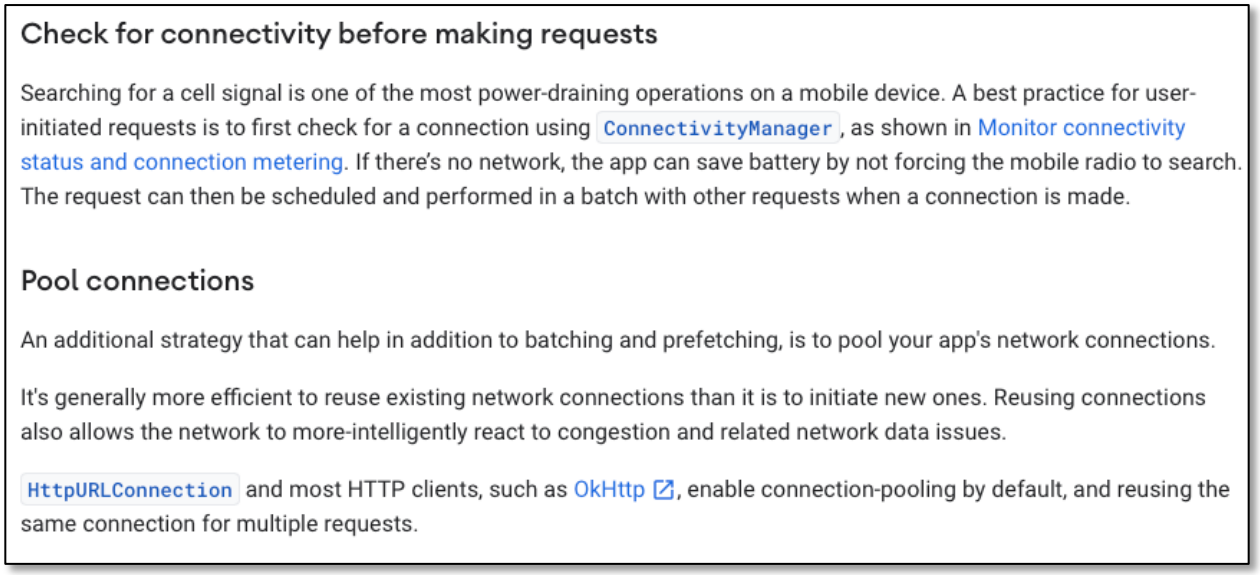
Of course, this is not always possible to do if your app needs to receive or send data immediately in response to a user action. You can mitigate this by anticipating and prefetching data. Other scenarios, such as sending logs or analytics to a server and other, non-urgent, app-initiated data transfers, lend themselves very well to batching and bundling. See Optimizing app-initiated tasks for tips on scheduling background network transfers.

Prefetch data

Prefetching data is another effective way to reduce the number of independent data transfer sessions that your app runs. With prefetching, when the user performs an action in your app, the app anticipates which data will most likely be needed for the next series of user actions and fetches that data in a single burst, over a single connection, at full capacity.

By front-loading your transfers, you reduce the number of radio activations required to download the data. As a result, you not only conserve battery life, but also improve the latency, lower the required bandwidth, and reduce download times.

Prefetching also provides an improved user experience by minimizing in-app latency caused by waiting for downloads to complete before performing an action or viewing data.

Claim	Public Documentation
	
18. The wireless end-user device of claim 1, wherein the differential traffic control policy defines that the first one or more applications can only access a first one of the network types during particular time windows.	The Accused Instrumentalities comprise "[t]he wireless end-user device of claim 1, wherein the differential traffic control policy defines that the first one or more applications can only access a first one of the network types during particular time windows." *See*, for example, the disclosures identified for claim 1.
19. The wireless end-user device of claim 1, wherein the one or more processors are configured to classify that the particular application is interacting with the user in the device user interface foreground based on a state of user interface priority for the application.	The Accused Instrumentalities comprise "[t]he wireless end-user device of claim 1, wherein the one or more processors are configured to classify that the particular application is interacting with the user in the device user interface foreground based on a state of user interface priority for the application." *See*, for example, the disclosures identified for claim 1.

Claim	Public Documentation
20. The wireless end-user device of claim 1, wherein the second one or more applications are not subject to a differential network access control that is applicable to the first one or more applications.	The Accused Instrumentalities comprise "[t]he wireless end-user device of claim 1, wherein the second one or more applications are not subject to a differential network access control that is applicable to the first one or more applications." *See*, for example, the disclosures identified for claim 1.
21. The wireless end-user device of claim 1, wherein the one or more processors are further configured to classify between: user applications; system applications, utilities, functions, or processes; and operating system application, utilities, functions, or processes.	The Accused Instrumentalities comprise "[t]he wireless end-user device of claim 1, wherein the one or more processors are further configured to classify between: user applications; system applications, utilities, functions, or processes; and operating system application, utilities, functions, or processes." *See*, for example, the disclosures identified for claim 1.
22. The wireless end-user device of claim 1, wherein the second one or more applications or services comprises foreground services.	The Accused Instrumentalities comprise "[t]he wireless end-user device of claim 1, wherein the second one or more applications or services comprises foreground services." *See*, for example, the disclosures identified for claim 1.
23. The wireless end-user device of claim 1, wherein selectively deny comprises intermittently block when the one or more Internet service activities are requested during selected time windows.	The Accused Instrumentalities comprise "[t]he wireless end-user device of claim 1, wherein selectively deny comprises intermittently block when the one or more Internet service activities are requested during selected time windows." *See*, for example, the disclosures identified for claim 1.
24. The wireless end-user device of claim 1, wherein the one or more processors are configured to pre-	The Accused Instrumentalities comprise "[t]he wireless end-user device of claim 1, wherein the one or more processors are configured to prevent the first one or more applications from changing the power state of at least one of the modems, and to not prevent the second one or more applications from changing the power state of the same modem or modems."

Claim	Public Documentation
vent the first one or more applications from changing the power state of at least one of the modems, and to not prevent the second one or more applications from changing the power state of the same modem or modems.	*See*, for example, the disclosures identified for claims 1 and 17.